Chapter 5

Multilevel Modeling

5.1. Introduction

We will first attempt to position multilevel model optimization within the more general framework of MDO. As detailed in Chapters 8 (theory) and 14 (algorithmic aspects), in the general MDO approach, optimization algorithms and simulation models appear to be decoupled: we will proceed with this assumption, while being aware that within the framework of specialized methods the problem can be solved by resorting to a specialized optimization algorithm taking the best possible advantage of the specificities of the problem [DES 07].

Having said that, the multilevel model optimization issue can be repositioned within the MDO framework. This comes down to substituting, in the simulation phase, a model by a series of models with increasing refinement.

It can be easily understood that carrying out a complete optimization (whether multidisciplinary or not) with a highly refined model can rapidly lead to prohibitive computing costs. Conversely, if a relatively coarse model is used, an optimum can still be obtained, but with little confidence in the results.

Thus, it appears to necessary to be able to vary the degree of refinement of a model throughout the optimization process in order to make this process both accurate and affordable in terms of computing cost. However, such a multilevel model optimization process does raise numerous questions:

Chapter written by Pierre-Alain BOUCARD, Sandrine BUYTET, Bruno SOULIER,Praveen CHANDRASHEKARAPPA and Régis DUVIGNEAU.

– When should the switch be made between model levels during the optimization process?

– How are optimization results transferred from one model level to another?

– Is it possible to validate the multilevel model optimization process *a posteriori*?

In the following discussion, using results from the literature, we will try to answer these questions and obtain a global picture of the main multilevel model optimization approaches available. This will enable us to propose original and innovative multilevel methods.
Let us note that this bibliographical study focuses on structural analysis, but some references are also available in the field of fluid mechanics.

5.2. Notations and vocabulary

5.2.1. *Notations*

We consider a functional J to be minimized with respect to k sets of n variables $(x_{(1)}, x_{(2)}, ..., x_{(n)})$ under equality constraints f and inequality constraints g. First, let us assume that the basic, *a priori* complex problem is decomposed into different model levels, with each higher level corresponding to a more refined model.

Then, let us introduce the different optimization levels and associated variables (see Figure 5.1). Here, we consider only two levels, i and $i + 1$, and we assume that the accuracy of the model increases with greater values of i. Let $(x_{(1)}, ..., x_{(i)})$ and $(x_{(i+1)}, ..., x_{(n)})$ denote the optimization parameters (commonly called design variables) of levels i and $i+1$, respectively. If the different levels are likely to exchange data other than the design variables, let $r_{(i)}$ denote the data transferred from level $i + 1$ to level i, and $t_{(i+1)}$ the data transferred from level i to level $i + 1$.

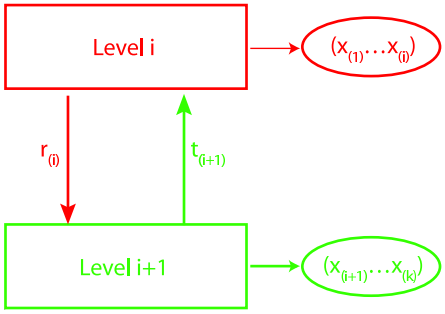


Figure 5.1. *General notations.*

5.2.2. *Vocabulary*

5.2.2.1. *Multidisciplinary and multilevel optimization*

In our introduction, we placed multilevel optimization within the scope of multidisciplinary optimization. Let us complete this initial framework by drawing a clear distinction between the two. In order to do that, we will refer to the work of [ENG 04], which presents these two types of optimization.

The purpose of the study is to optimize the flight scenario of an airplane. In this context, different phenomena pertaining to different disciplines (structure, flight mechanics, aerodynamics, etc.) are involved. Obviously, the optimization of the flight scenario is a problem which couples all these disciplines, and so does the resulting optimum. However, we will see that the proposed study, which falls into the category of **multidisciplinary** optimization, does not actually couple these disciplines, or that it does so only partially.

Within each of the different disciplines (see Figure 5.2), one or several modeling levels are proposed.

For example, with regard to the "structure" discipline, we seek to minimize the overall mass of an airplane wing, taking into account multiple constraints and structural details. Since the optimization of such a model is still too risky and costly in terms of computing time, we decompose this optimization into two different levels. The **multilevel** strategy consists in defining a somewhat coarse basic model, then building increasingly refined "children" models. Figure 5.2 [ENG 04] represents the different disciplines to be studied for multidisciplinary optimization along with the different levels of study to be considered for multilevel optimization. Thus, within the "structure" discipline, the first level (called the intermediate level in Figure 5.2) corresponds to the complete "airplane wing" model. At the second level (called the subsystem level), the zone of interest, which contains a detail, is modeled to a higher degree of refinement.

Finally, coupling among the different disciplines is achieved relatively simply: the data resulting from the "aerodynamics" optimization are used in the "structure" optimization; similarly, the mass of the airplane resulting from the "structure" optimization is necessary for the "flight mechanics" optimization. It is important to note that this is not a study of the complete system, but the optimization of the decoupled complete system in the sense that the influence of structural modifications on the aerodynamic calculation, for example, are not taken into account.

In [ENG 04], the multidisciplinary/multilevel distinction is clear: several disciplines are considered, and some of them are optimized using different modeling levels.

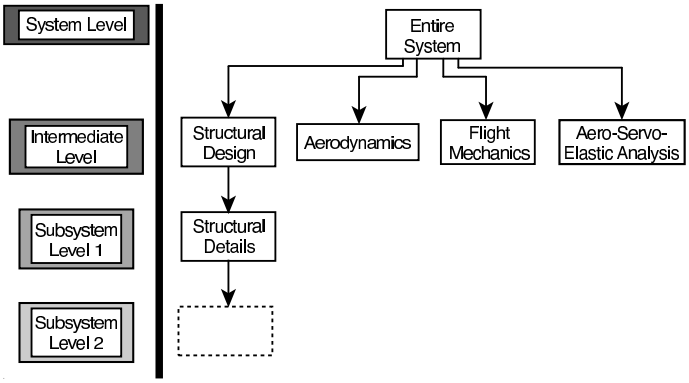


Figure 5.2. *Multidisciplinary and multilevel optimization.*

Unfortunately, this distinction is not made by all authors in the literature.

Indeed, in [CHA 95, CHE 05], these two types of optimization are merged or, more exactly, the concept of level is tied directly to the concept of discipline. For example, in [CHA 95], we find an approach to the dimensioning of the blades of a helicopter developed over three levels. The first level deals with the "aerodynamics" performance of the blades; at the second level, the "dynamic" characteristics of the rotor are determined in order to reduce the stresses in the blades; finally, at the third level, an attempt is made to reduce the mass of the "structure".

Therefore, we consider it important to clarify the different interpretations which can be made of multilevel model optimization. First, in order to make our discussion easier to understand, we will consider a single discipline. Thus, sharing the point of view of [ENG 04], we will eliminate the fact that the terms *discipline* and *level* have the same meaning. Nevertheless, the methods presented remain suitable for numerous disciplines, which puts us *de facto* in an MDO context.

Even though the following presentation is limited to the study of a single discipline, we will see that multilevel model optimization still encompasses two major categories of strategies:

– strategies in which the term *multilevel* refers to the optimization process, but which rely on a single model (Chapter 6);

– strategies in which the term *multilevel* refers to the model itself, in which case one or several optimization processes use the different modeling levels (Chapter 3).

5.2.2.2. *Multilevel optimization: different meanings*

The term *multilevel optimization* tends to be used excessively to indicate that the problem being addressed is relatively complex and, therefore, is treated in several steps (i.e. on several levels). Consequently, the expression "multilevel optimization" has quite different meanings in different studies. Nevertheless, we can extract two main definitions from the literature (directly associated with the two strategies introduced previously).

– In what is classically called **multilevel parameter optimization**, the initial model remains unchanged throughout the optimization process, but is treated with different accuracy requirements. Since this topic is largely discussed in Chapter 6, section 5.4 will merely present some illustrations taken from structural analysis.

– In **multilevel model optimization** (an expression we will use from now on to designate what constitutes the core of the approach we are proposing), the initial model is allowed to evolve during the optimization process. Two main types of enrichment can be found in the literature (Figure 5.3):

- *Hierarchical descriptions*, presented in section 5.5.1, is based on consideration of successive models of increasing refinement. In general, the necessary information is transferred only from level i to level $i + 1$, thus enabling us to "forget" the most primitive model permanently: for example, a more or less empirical analytical functional, followed by a beam model, then a linear 3D model, finally a non-linear 3D model. This is typically the approach used in, for example, a global–local analysis.

- In *imbricated descriptions*, developed in section 5.5.2, the basic model is always preserved and is actually enriched from one level to the next, thus enabling the transfer and recovery of information among the different levels. This corresponds to a multiscale description of the problem which is capable of taking into account the different modeling levels (or scales) *simultaneously*, and which can be enriched throughout the optimization process.

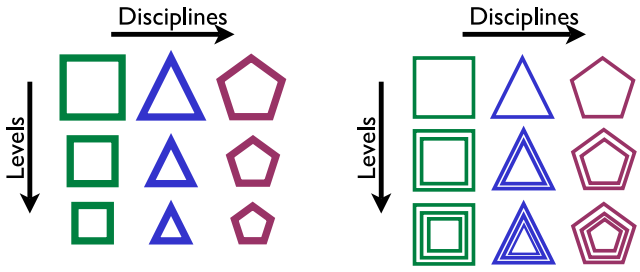


Figure 5.3. *Hierarchical models (left) and imbricated models (right).*

Finally, although most existing strategies fall into one of the two categories presented above, we can also find in the literature, described as “multilevel optimization”, problems decomposed into several subproblems (e.g. using a domain decomposition method), which are optimized according to the same criterion, either independently from one another or in a coupled scheme. In general, these strategies are used for the resolution of problems with very large numbers of degrees of freedom. We will designate such strategies as **parallel model optimization methods**.

In summary, we can distinguish three main groups of methods, all labeled as “multilevel”, which we categorize into:

– parallel model optimization;

– multilevel parameter optimization;

– and multilevel model optimization.

5.3. Parallel model optimization

Although this type of method is quite different from the multilevel model optimization we are going to use, we will present its main governing principles in the case of structural analysis problems. Note that by analogy with the notations introduced previously each “level” will be associated with a subdomain, even though all levels are completely equivalent in terms of the degree of refinement of the description.

These parallel methods are essentially based on domain decomposition methods, which provide an elegant means of parallelizing the resolution of a problem defined on a structure. Two main application frameworks can be envisaged:

– the case in which the calculation of the cost function alone is parallelized (Figure 5.4);

– the case in which both the calculation of the cost function and the optimization process are parallelized (Figure 5.5).

An example of application of the first type of method can be found in [ELS 91]. This work primarily focuses on the optimization of the cost function by distributing the calculations in the subdomains among different processors, while minimizing communications among the processors. Indeed, it is the communications and the balance among the processors which most influence the speedup.

An example of the use of the second type of strategy is given in [UME 05]. In this work, the uncoupling of the optimization tasks allocated among the processors is facilitated by the fact that only prescribed displacement-type quantities are necessary for carrying out the optimization on each processor.

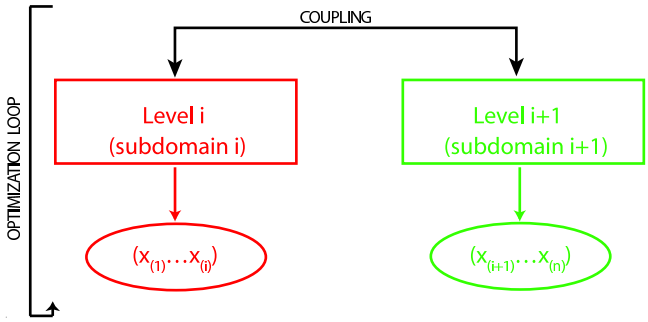


Figure 5.4. *Parallel optimization (parallelization of the cost function).*

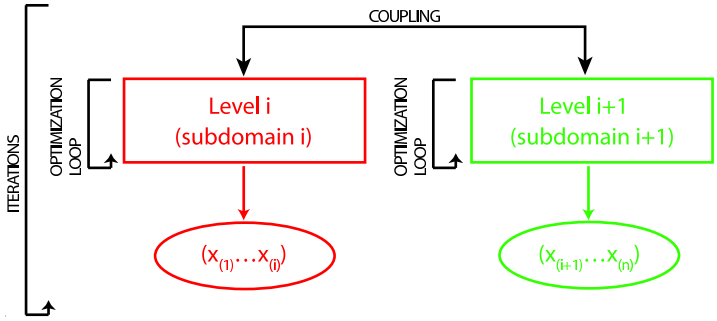


Figure 5.5. *Parallel optimization(parallelization of both the cost function and the optimization process).*

Thus, at each iteration, the stiffness matrix of each substructure must be updated in order to determine the prescribed displacements which, when applied to the substructures, enable the optimization phase to be carried out under given boundary conditions. Iterations are then necessary to ensure that the optimization loops in each processor contribute towards the global convergence of the optimization process.

5.4. Multilevel parameter optimization

In this type of method, the model is well-defined right from the first level of calculation and, thus, the model and the cost function J remain the same at all optimization levels. The optimization problem is defined classically as finding an optimum of J in the space of the variables $(x_{(1)}, x_{(2)}, ..., x_{(n)})$, with the possible addition of equality constraints f or inequality constraints g.

The multilevel optimization process consists in transforming the problem:

$$min_{x_{(1)},...,x_{(n)}} \quad J(x_{(1)}, ..., x_{(n)})$$

into a series of problems

$$min_{x_{(i)},...,x_{(j)}} \quad J(x_{(1)}, ..., x_{(n)})$$

$$\text{with } (x_{(1)}, ..., x_{(i-1)}, x_{(j+1)}, ..., x_{(n)}) \text{ fixed.}$$

This is nothing but the application of the simple idea that, rather than optimizing by taking into account all the design parameters simultaneously, we should first minimize the cost function over only a reduced number of variables, typically $(x_{(i)}, ..., x_{(j)})$. We then enrich the list of design variables during the optimization process, for example by taking $(x_{(i)}, ..., x_{(j)}, , x_{(n)})$. Upon each enrichment of this list, the optimization process can be carried out in two different ways:

– by solving the optimization problem over the whole set of the current design variables $(x_{(i)}, ..., x_{(j)}, , x_{(n)})$ (a technique called sequential optimization);

– or by solving the optimization problem by fixing the design variables $(x_{(i)}, ..., x_{(j)})$ determined in previous steps, and allowing changes only in the new variables $(x_{(j+1)}, ..., x_{(n)})$ (a technique called iterative optimization).

5.4.1. *Sequential optimization*

The main distinctive feature of this method is that it relies on a classification of the design variables according to their importance. This classification can be achieved by starting with the most global variables, such as the overall dimensions of the structure, and moving towards the most local variables, such as geometric parameters describing structural details. Another method of classification would be to use sensitivity analysis and sort the design variables from the most influential to the least influential.

For example, assuming that the parameters $(x_{(1)}, ..., x_{(i)})$ are the dominant variables in the optimization, the first optimization loop is carried out using these variables. We then increase the number of design variables progressively by adding, at a child's level, the variables $(x_{(i+1)}, ..., x_{(n)})$ to the parameters $(x_{(1)}, ..., x_{(i)})$. A schematic representation of this method is given in Figure 5.6.

A typical example of this procedure can be found in [KRA 05]. The problem considered is the optimization of the mass of a lattice-type structure whose space of design variables contains the topological parameters, the material data, the geometric parameters of the bars, and the connections among the bars.

At the first calculation level, the optimization is performed only on the topological parameters of the structure.

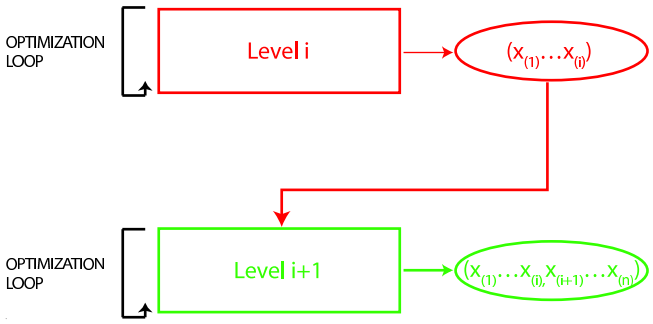


Figure 5.6. *Sequential optimization of multiple levels of parameters.*

$$min_{x_{(1)}} J(x_{(1)}) \text{ with } (x_{(2)}, x_{(3)}, x_{(4)}) \text{ fixed}$$

After convergence, the material data $x_{(2)}$ are added to the topological parameters $x_{(1)}$, leading to the second optimization level.

$$min_{x_{(1)},x_{(2)}} J(x_{(1)}, x_{(2)}) \text{ with } (x_{(3)}, x_{(4)}) \text{ fixed}$$

At the third and fourth levels, the list of design parameters is increased again by adding the standard geometric parameters and the connections among the bars. At the last level, the minimization of the cost function (i.e. the mass) can be expressed as:

$$min_{x_{(1)},x_{(2)},x_{(3)},x_{(4)}} J(x_{(1)}, x_{(2)}, x_{(3)}, x_{(4)})$$

Additional illustrations of this strategy can be found in [KRA 03] or [LIU 04].

5.4.2. *Iterative optimization*

In this case, the first-level optimization only concerns variables $(x_{(1)}, ..., x_{(i)})$, all other variables being fixed. Once convergence has been reached, the optimum values of $(x_{(1)}, ..., x_{(i)})$ obtained are retained for the calculation at level 2, which concerns parameters $(x_{(i+1)}, ..., x_{(n)})$. During the calculation, each successive level is optimized iteratively, knowing the results from the levels already treated (see Figure 5.7). Thus, after convergence, optimal solution $(x_{(1)}, ..., x_{(n)})$ is obtained.

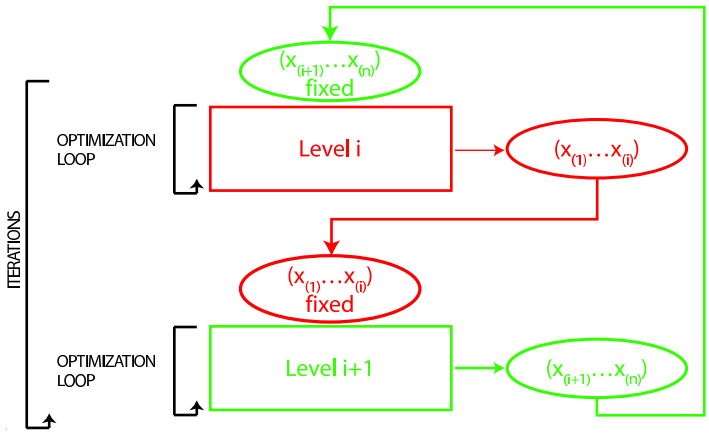


Figure 5.7. *Iterative optimization for multiple levels of parameters.*

The multidisciplinary and multilevel optimization of [CHE 05] belongs completely to this type of resolution algorithms. In that work, the optimization concerns a structure consisting of an assembly of bars. The optimization criterion takes into account, on the one hand, a criterion of the dynamic behavior of the structure (assuming that the bars are rigid solids) and, on the other hand, a criterion of the sizing of consecutive bars in the mechanism. At the first level, the different parameters of the dynamic behavior $x_{(1)}$ are optimized, while keeping the dimensioning parameters $x_{(2)}$ fixed so the structure can have maximum freedom of movement.

$$min_{x_{(1)}} J_1(x_{(1)}, x_{(2)}) \text{ with } x_{(2)} \text{ fixed}$$

On the second level, the optimum parameters $x_{(1)}$ resulting from the previous level are introduced and kept constant, and the design variables $x_{(2)}$ associated with the sizing of the bars are optimized in order to minimize the mass of the structure.

$$min_{x_{(2)}} J_2(x_{(1)}, x_{(2)}) \text{ with } x_{(1)} \text{ fixed}$$

If the procedure were interrupted after this first step, we could consider the optimization problem to be solved by assuming that the two levels of optimization are decoupled, which is clearly not true in this case. (However, we could think of applications for which this would be possible, see for example [LER 98], in which one of the methods proposed takes advantage of the independence of the in-plane behavior and out-of-plane behavior of a composite material.)

In order to couple the two levels, we must successively iterative on level 1 and level 2. Numerous illustrations of this iterative multilevel optimization procedure are available.

For example, [THE 98] concerns the optimization of a composite structure, taking its microstructure into account. Since a finite element calculation involving the whole microstructure is inconceivable, an intermediate non-linear calculation phase of an elementary cell of the composite is used. (An alternative solution, proposed in [BEN 95], consists in assuming that the elementary cell behaves linearly and treating it analytically through homogenization.) Then, the design variables are related to the microstructure $x_{(1)}$ (fiber volume fraction, mechanical properties of the fibers and of the matrix) and to the density of the material $x_{(2)}$ which is involved in the structural analysis in order to minimize the mass. Thus, this optimization problem can be handled as follows:

– At level 1, we fix the density $x_{(2)}$ and determine the parameters $x_{(1)}$ which minimize the flexibility of the elementary cell.

– At level 2, given the variables $x_{(1)}$, we determines the value of $x_{(2)}$ which minimizes the mass of the structure. This optimization phase does not use $x_{(1)}$ directly, but uses the equivalent stiffness resulting from the calculation of the elementary cell: nevertheless, the variables $x_{(1)}$ are fixed because the stiffness is fixed.

We will mention, as the last illustration, [CON 02] concerning the optimization of a multiple ply composite beam structure, taking into account geometric nonlinearities. Again, in this case, the algorithm used is of the iterative type with two levels, and:

– The first-level optimization consists in maximizing the critical buckling load of the ply, with the design variable being the orientation of the plies $x_{(1)}$, the geometric variables $x_{(2)}$ (thickness of the plies, width and thickness of the beam) being fixed to prevent the critical load from being exceeded.

– Level 2 concerns the minimization of the mass of the structure, with the orientation of the plies $x_{(1)}$ being fixed, the design variables $x_{(2)}$ being the thickness of the plies and the dimensions of the beam.

iT should be noted that in the proposed method the optimization algorithm used at each level is a genetic algorithm.

Other illustrations of the use of this strategy can be found in [LER 98] (optimization of a stack of composite plies with the total number of plies, their orientation, and the stacking sequence as the design variables).

For the two proposed methods, the optimality of the solution can be related implicitly to the strategy used and, more specifically in this case, to the convergence of the "fixed-point" type of algorithm used: had the sequential optimization method been used, it is quite possible that the solutions resulting from the two algorithms would have been different, even although the initial problem is the same. This remark is valid whether we optimize a single criterion or several criteria. In the case of a single criterion, the "optimum" solutions obtained using the different algorithms can be different. In the case of a multicriteria optimization – in which there is no optimum solution, but a unique Pareto front – the different methods can generate different Pareto fronts.

Therefore, although numerous sequential or iterative strategies have been developed, they do not all converge towards the optimum solution of the initial problem (see in [ALE 00]).

5.5. Multilevel model optimization

Unlike the methods described in section 5.4, where the model remained unchanged from one optimization level to the next, the methods we are about to present concern the optimization of several more or less detailed models, among which we can freely choose which one to use at any time during the optimization process. This is therefore indeed a process in which several model levels are available throughout the optimization.

Two cases can be distinguished, depending on the way these models coexist: hierarchical multilevel models and imbricated multilevel models.

5.5.1. *Hierarchical optimization*

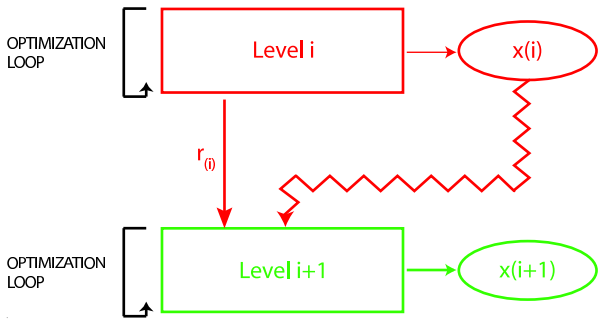


Figure 5.8. *Hierarchical optimization.*

Here, we are dealing with a context in which the model actually undergoes changes during the different optimization stages (for example changes in behavior, different meshes, or even different models). Then, the data acquired at level i are transferred in order to initialize the optimization at level $i + 1$.

In addition to a number of parameters $x_{(i)}$, which are transferred as they are or "adapted", various data $r_{(i)}$ are communicated from level i to level $i + 1$ in order to achieve the best formulation possible for the new model (see Figure 5.8). By "adapted", we mean that the transfer of data may not be performed explicitly, but can involve a separate calculation, such as a projection.

The optimization of level $i + 1$ is carried out completely independently of those of the higher levels, on which, consequently, it has no influence. We can say that the model on level i is permanently "forgotten". A consequence of this is that information recovery is problematic, even non-existent.

The fact that information cannot be traced back from level $i + 1$ to level i remains insignificant as long as the influence of the level $i + 1$ model on level i is negligible. While it is sometimes possible to get information *a priori* concerning this influence, this is not always the case, and a kind of "butterfly effect" during the optimization process would make this process completely worthless.

In [ROB 99], the objective is to try to minimize the drag of an airplane wing, taking into account a number of constraints. Three different models ranging over three levels from the most empirical to the most realistic are used. In that particular case, this progressive refinement is reflected in the finite element mesh of the wing and in the modeling of the fluid behavior.

This example is a typical case of what is called a hierarchical method in the sense that the program used for optimization i is different from that used for optimization $i + 1$, which implies a complete and irreversible transfer of information. [KEA 00] discusses a similar process applied to the design of an airplane wing.

5.5.2. *Imbricated optimization*

The choice made in this case is radically different from that presented previously: as in the hierarchical strategy, the quality of the model is enhanced with each increase in level, but the models communicate with one another in both directions within the optimization process. Thus, the level $i+1$ optimization influences the level i optimization, and *vice versa*.

Indeed, not only do we transfer data $r_{(i)}$ as in all other methods, but, in addition, we can recover data $t_{(i+1)}$, as shown in Figure 5.9. This bidirectional transfer of data

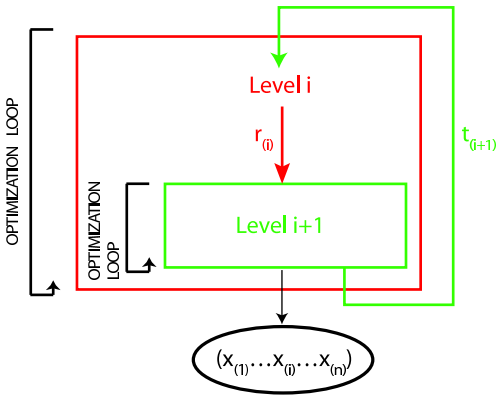


Figure 5.9. *Imbricated optimization.*

requires the models to be capable of communicating with one another. (For example, the transfer between a beam model and a 3D model can be achieved, in one direction, by a stress distribution assumption and, in the other, by integration.) However, this communication can be established much more naturally within the framework of a multiscale strategy.

This is what makes this method perfectly suited to the multiscale description of a problem, a description which enables us to model the different scales of the same structure in parallel, taking into account the influence of the optimization on the refined scale over the optimization at the global level.

Let us go back to [ENG 04], which has already been presented in section 5.2, because it gives a good illustration of imbricated optimization and can be easily generalized to other problems. This time, we are considering the "structure" discipline alone, in which the complete problem is decoupled on two levels, represented in Figure 5.10. At level 1, we consider the whole model of the airplane wing containing one or more superelements. At level 2, we consider the more refined model of a superelement containing a structural detail (in this case, a door).

The optimization concerns the total mass of the airplane wing under the strict constraint that the von Mises' elementary stresses σ (which are functions of the second invariant of the deviatoric stress tensor alone) remain less than a prescribed limit stress. The multilevel optimization problem can be expressed as:

– Level 1

$$min_{x_{(1)}} \quad J_1(x_{(1)}, x_{(2)})$$
$$\text{with } x_{(2)} \text{ fixed}$$

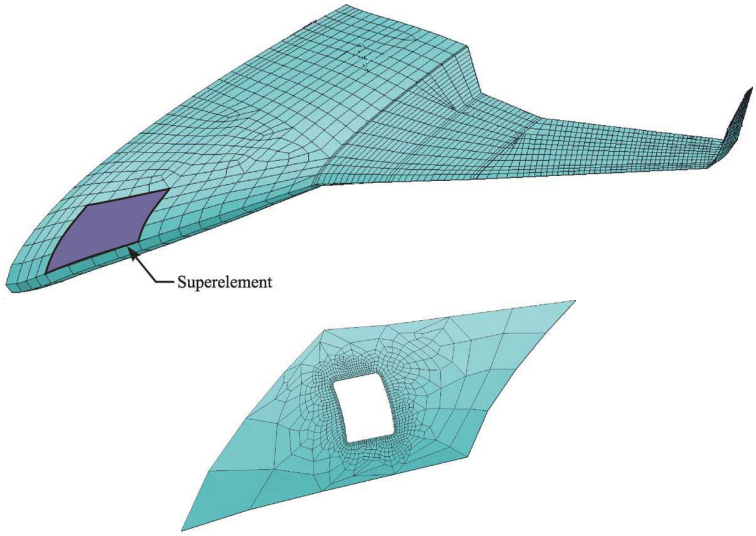


Figure 5.10. *Level* 1*: airplane wing; level* 2*: superelement.*

$$\text{knowing that } \begin{cases} g_1(x_{(1)}, x_{(2)}) < 0 \\ g_{2_{app}}(x_{(1)}, x_{(2)}) < 0 \end{cases}$$

– Level 2

$$min_{x_{(2)}} \quad J_2(x_{(1)}, x_{(2)})$$

$$\text{with } x_{(1)} \text{ fixed}$$

$$\text{knowing that } \begin{cases} g_{1_{app}}(x_{(1)}, x_{(2)}) < 0 \\ g_2(x_{(1)}, x_{(2)}) < 0 \end{cases}$$

where $g_{1_{app}}$ (respectively $g_{2_{app}}$) is the inequality constraint associated with level 1 (respectively 2) approximated during the optimization calculation of level 2 (respectively 1).

In other words, the stresses at level 1 take into account the stresses from another level. Indeed, in the optimization process, we optimize level 1, and for each calculation step of the cost function associated with level 1 it is necessary to know the optimum model at level 2 (which also implies an optimization process).

Thus, the stresses at the two levels are coupled. Due to the imbrication of the models, the stresses at one level involve an approximation of the stresses at the other.

It is only after convergence of the whole optimization process that the stresses at both levels cease to be approximations.

It is interesting to observe that the introduction of the optimization processes at both levels enables the natural coupling of the two modeling levels: in a way, we take advantage of the iterations of the optimization algorithm to couple the modeling levels.

Finally, the calculation and optimization strategy comes down to the following:
Beginning of the optimization process at level 1

– The stiffness of the superelement is fixed. The calculation is carried out at level 1 in order to evaluate the cost function J_1 (the mass of the airplane wing) with the inequality constraints g_1 and $g_{2_{app}}$. Therefore, we must determine $g_{2_{app}}$.

– At each stage of the optimization, constraint g_1 is approximated by $g_{1_{app}}$, which is transferred to level 2 along with the loading and the boundary conditions.

$$r_{(1)} = \left\{ \begin{array}{c} g_{1_{app}} \\ \text{loads} \\ \text{displacements} \end{array} \right\}$$

– The boundary conditions of the superelement are known. We perform the optimization of the mass J_2 of the superelement with the constraints g_2 and $g_{1_{app}}$. Thus, variables $x_{(2)}$ are determined.

– We transfer a stiffness corresponding to the superelement along with approximation $g_{2_{app}}$ of constraint g_2.

$$t_{(2)} = \left\{ \begin{array}{c} g_{2_{app}} \\ \text{stiffness of the superelement} \end{array} \right\}$$

– The optimizer updates the design variables $x_{(1)}$ taking into account g_1 and $g_{2_{app}}$.

End of the optimization process on level 1

This method is both sound and efficient. Nevertheless, because of the two imbricated optimization loops, it requires significant computing times.

Although[TOS 06] does not belong to the framework of imbricated model optimization, it is useful to comment briefly on the general optimization algorithm proposed. Indeed, an interesting aspect of the study concerns the transfer of data in both directions between two consecutive levels. In order to do this, we must prescribe that the data being sent $r_{(i)}$ and the data being received $t_{(i+1)}$ between one level and the other be identical. Thus, one of the indispensable stopping criteria of the iterative process is that the data $r_{(i)}$ and $t_{(i+1)}$ be identical for both levels. This condition requires the integration of a new constraint (in addition to the classical r and t) into both calculation levels i and $i + 1$. Finally, it is proposed to take this constraint into account in the optimization of the cost function through the updated Lagrangian method.

Taking into consideration the previous statements, we propose a new and efficient method for multilevel model optimization based on a multiscale modeling which requires only a single global optimization loop. We will illustrate this method in the context of dealing with structural details: typically, the position of a hole in a structure, the shape of the contact surfaces between different parts, the tightening parameters in an assembly.

5.6. General resolution strategy

Structural optimization involves two very different domains of expertise: numerical simulation and optimization. Starting from a set of design variables x obtained from the optimizer, the simulator returns the corresponding objective function(s) and possibly the constraint functions. Then, the optimization process tends to minimize this objective (or these objectives). There are two ways to optimize a mechanical model.

The optimization is carried out using en empirical model, called a *metamodel*, generated from a number of measurement points (evaluated by the simulator). These points can be positioned logically, i.e. regularly (response surface technique [ROU 98]), or chosen randomly, as in the kriging method [SAK 03]. In fact, the choice of the simulation points (e.g. through design of experiments, Latin hypercube sampling, random drawing) is generally independent of the metamodel built from these points (polynomial response surfaces, kriging, RBF networks, etc.). This approach is used in the methodology developed in Chapter 3.

The optimization can also be performed directly using the mechanical model. In this case, the solution obtained is exact and the two fields of expertise exchange data (design parameters, constraints, and objective functions). Sometimes, the simulation and the optimization are performed in the same program, and the exchange of data takes place naturally. In our case, the two disciplines are decoupled; the two programs proceed in parallel and exchange information through data files. This uncoupling allows great freedom in the choice of approaches and algorithms, and the highest possible expertise in each of the two domains of application.

Figure 5.11 represents the complete optimization scheme, in which the metamodel and the mechanical model are optimized successively.

We are now going to propose a high-performance numerical simulation method for multilevel optimization. This technique can be applied to both stages of the optimization strategy and concerns the gray-colored simulation steps in Figure 5.11. Usually, these simulation steps are very costly in terms of CPU time because they are repeated

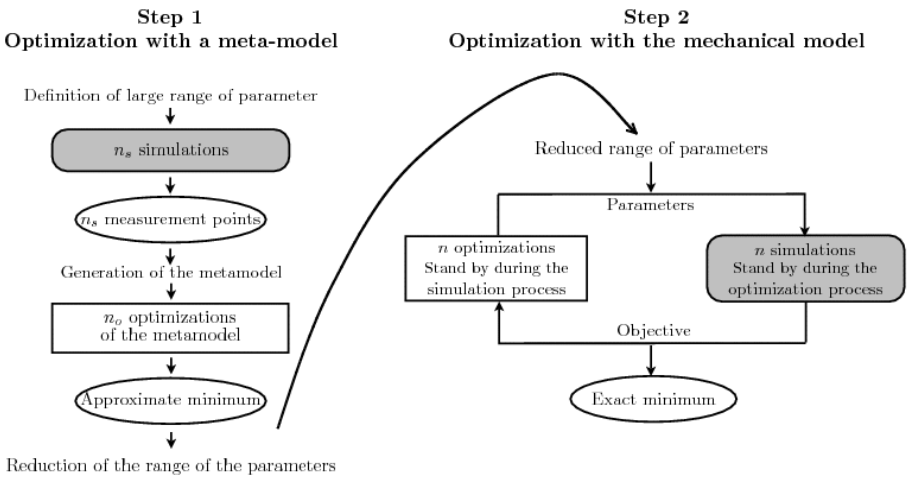


Figure 5.11. *The complete optimization scheme.*

$n_s + n$ times. The multiresolution strategy enables reuse of the results of similar calculations in order to reduce computing time.

The simulation strategy adopted is very well suited to the treatment of these two costly stages. More specifically, the multiresolution aspect is a fundamental point in the reduction of computing costs, with reard to both the calculations required for the construction of the metamodel and the optimization of the complete model.

Our approach is based on two fundamental points.

– The first point is the use of a *micro–macro* resolution technique, which has already been proved to be remarkably efficient numerically [LAD 02]. This technique is a *multiscale* method based on two features:

- decomposition of the domain into substructures and interfaces with their own unknowns. One of the advantages of the proposed decomposition is that the unknowns at the interfaces are mixed quantities (loads and displacements). The multiscale aspect of the method is introduced through the quantities defined at the interface, which are divided into a “macro” part (e.g. resultants and moments) and a “micro” part (the complementary part).

- the LATIN method (described in detail in section 5.7.1.3), which is a general resolution strategy for non-linear problems. In particular, in the context of domain decomposition, this method enables problems associated with the different substructures to be solved independently and this information to be transferred across the interfaces.

– The second fundamental point is the introduction of an appropriate strategy for the description of geometric details such as fillets or holes. We choose a mesh-independent technique combining a local enrichment method (X-FEM) and the use of level set functions which enable us to "activate" the integration of the structural detail without modifying the mesh during the optimization process. In the case of details such as contact surfaces or fasteners, it is unnecessary to use the X-FEM because these details can be dealt with directly at the interface level as they do not influence the mesh.

The proposed approach can also be applied without introducing a multiscale aspect and, even in this case, it can be very efficient [BOU 03, BOU 04]. In fact, some of the examples presented were calculated with a single-scale version of this approach.

These different techniques have already been used in optimization processes, for example the use of the single-scale version of the LATIN method in the context of an identification problem [ALL 05b]. The description using level set functions was used in an optimization context in [WAN 03, ALL 05a].

By analogy with our previous presentation, we can consider that the global calculation over the whole domain (the macro level) corresponds to the calculation at level 1, and that the calculation in the substructures (the micro level) corresponds to the calculation at level 2.

Since the macro problem and the micro problem are coupled within the LATIN resolution method, bilateral communication between the two levels ceases to be a problem and can be established naturally through the interfaces. Thus, the optimization process can be considerably simplified by using a single optimization loop (see Figure 5.12). However, it is also possible to use two optimization loops, in which case the scheme resembles that of [ENG 04], except for the interaction between the levels. Thus, depending on the problem, one can take advantage ofwe can use either a single or a double optimization process.

We will also see that another advantage of our proposed strategy is that it can reuse calculations previously done for one set of design variables $(x_{(1)}, x_{(2)}, ..., x_{(n)})$ to perform another calculation associated with new values of the design variables. This property is called "multiresolution", in the sense that it enables the multiple resolution of problems parametrized by the design variables to be handled at minimum cost [BOU 03, BOU 04].

In order to position this new approach among those presented before, this optimization is viewed as a multilevel model optimization because the model is enriched from one level to the next (typically, the structural detail is treated at level 2), and the

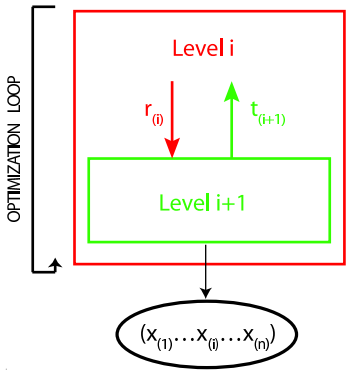


Figure 5.12. *Optimization using the multiscale approach.*

optimization is clearly imbricated because the communication between the levels is established naturally within the resolution algorithm.

This is especially true in the case of the multiscale version of the method; indeed, two modeling levels are introduced explicitly and communicate with one another during the resolution process. The macroscopic level is a homogenized version of the complete problem and, thus, constitutes a simplified modeling level of the complete model. The microscopic level comes up during the iterative resolution process with the LATIN method to complement the macroscopic level and finally solve the complete model.

In addition, if a metamodel is used for the first optimization phase, we can consider that we have carried out an optimization with three modeling levels: the metamodel, the macroscopic model, and the complete model.

5.7. Use of the multiscale approach in multilevel optimization

5.7.1. *The micro–macro approach*

The multiscale approach proposed here, introduced by [LAD 01] and called the "micro–macro" approach, is based on three fundamental points which are described in detail below.

5.7.1.1. *Domain decomposition*

Let us consider an elastic structure Ω subjected, under the small-perturbation assumption, to a loading $\underline{F}_d$ over a portion $\partial_2\Omega$ of its boundary $\partial\Omega$. Over the complementary part $\partial_1\Omega$, the displacement $\underline{U}_d$ is prescribed. The first point of the micro–macro approach consists in dividing the mechanical system into substructures and interfaces (Figure 5.13). Each of these constituents corresponds to a complete mechanical entity characterized by its own variables and its own behavior. This is a natural view when the substructures are the constitutive parts of an assembly and the interfaces correspond to the connections among these parts.

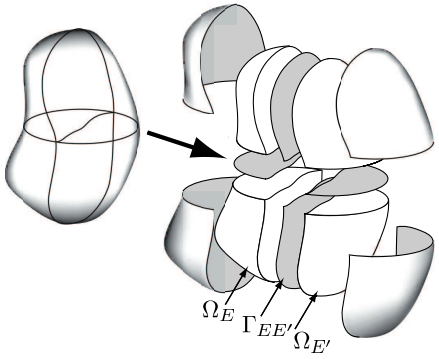


Figure 5.13. *Decomposition of the medium into substructures and interfaces.*

Let us now consider a substructure E defined in domain Ω_E bounded by $\partial\Omega_E$. E is subjected to the action of its environment (the neighboring interfaces) in the form of a load distribution $\underline{F}_E$ and a displacement distribution $\underline{W}_E$ (Figure 5.14). The interface $\Gamma_{EE'}$ between the two substructures Ω_E and $\Omega_{E'}$ expresses a behavior law between $(\underline{F}_E, \underline{F}_{E'})$ and $(\underline{W}_E, \underline{W}_{E'})$. The introduction of displacement distributions and interface load distributions gives this domain decomposition method a mixed character. Subsequently, we will address two types of problem: one associated with the substructures, and the other associated with the interfaces.

5.7.1.1.1. The substructure problem

Let $\underline{u}_E$ denote the displacement field at any point of substructure Ω_E, and ϵ_E and σ_E the corresponding strains and stresses, respectively.

The mechanical problem to be solved in each substructure consists in finding $\underline{u}_E$ and σ_E which verify:

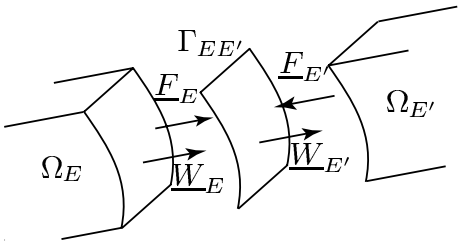


Figure 5.14. *Exchanges among substructures and interfaces.*

– kinematic admissibility:

$$\forall(\underline{u}_E, \underline{W}_E) \in \mathcal{U}_E \qquad \underline{u}_E|_{\partial\Omega_E} = \underline{W}_E \tag{5.1}$$

where $\mathcal{U}_E$ is the space of the kinematically admissible fields;

– equilibrium in a weak sense (assuming that the volume force distribution is identical to zero):

$$\forall(\underline{u}^*_E, \underline{W}^*_E) \in \mathcal{U}^*_E\,,\ \forall(\underline{F}_E, \sigma_E) \in \mathcal{S}_E \int_{\Omega_E} \sigma_E : \epsilon(\underline{u}^*_E) d\Omega - \int_{\partial\Omega_E} \underline{F}_E . \underline{W}^*_E = 0 \tag{5.2}$$

where $\mathcal{S}_E$ is the space of the statically admissible fields;

– the constitutive relation (assuming linear elastic behavior):

$$\sigma_E = \mathbf{D}\epsilon(\underline{u}_E) \tag{5.3}$$

where **D** is Hooke's operator.

The resolution of the problems associated with each substructure is performed independently of the other substructures and, thus, can be parallelized.

5.7.1.1.2. The interface problem

The mechanical problem to be solved over each interface consists in determining $\underline{W}_E$ and $\underline{F}_E$ which verify:

– static equilibrium:

$$\underline{F}_E + \underline{F}_{E'} = 0 \tag{5.4}$$

– the constitutive relation:

$$\mathrm{R}(\underline{W}_E, \underline{F}_E, \underline{W}_{E'}, \underline{F}_{E'}) = 0 \tag{5.5}$$

where R is the (linear or non-linear) behavior law of interface $\Gamma_{EE'}$, which we will develop in detail in section 5.7.2.

Bringing everything together, we define $\mathbf{s} = \sum_E \mathbf{s}_E$ as the solution of the substructured problem of the form $\mathbf{s}_E = (\underline{u}_E, \underline{W}_E, \sigma_E, \underline{F}_E)$ associated with each substructure, whose neighboring interfaces verify equations [5.1] to [5.5].

5.7.1.2. *The multiscale aspect at the interfaces*

Here, contrary to most multiscale approaches, the scale separation takes place at the interface level. Thus, each interface quantity is expressed as the sum of a macro part and a micro part: $\underline{F} = \underline{F}^M + \underline{F}^m$ and $\underline{W} = \underline{W}^M + \underline{W}^m$. The macro displacements $\underline{W}^M$ and macro loads $\underline{F}^M$ are sought in spaces $\mathcal{W}^M_{EE'}$ and $\mathcal{F}^M_{EE'}$, defined below. The macro components are obtained through a projector $\mathbf{\Pi}_{\Gamma_{EE'}}$ which verifies the following uncoupling relation between the micro work and the macro work:

$$\int_{\Gamma_{EE'}} \underline{F}.\underline{W} dS = \int_{\Gamma_{EE'}} \underline{F}^M.\underline{W}^M dS + \int_{\Gamma_{EE'}} \underline{F}^m.\underline{W}^m dS \qquad [5.6]$$

Then, we get: $\underline{F}^M = \mathbf{\Pi}_{\Gamma_{EE'}}(\underline{F})$ and $\underline{F}^m = (id - \mathbf{\Pi}_{\Gamma_{EE'}})(\underline{F})$. The macro and micro components of the displacement $\underline{W}$ are expressed in the same way.

In order to make the projection operator $\mathbf{\Pi}_{\Gamma_{EE'}}$ explicit, let $\mathbf{e}^M_{EE'} = (\underline{e}^M_1, .., \underline{e}^M_{n_M})$ denote a basis of space $\mathcal{F}^M_{EE'}$. Then, we have:

$$\underline{F}^M = \mathbf{\Pi}_{\Gamma_{EE'}}(\underline{F}) = \sum_{i=1}^{n_M} (\underline{F}, \underline{e}^M_i)\, \underline{e}^M_i = \sum_{i=1}^{n_M} [F^M]_i\, \underline{e}^M_i \qquad [5.7]$$

where $[F^M]_i$ represents the components of $\underline{F}^M$ in the macroscopic basis $\mathbf{e}^M_{EE'}$. Let us note that the macro components $[F^M]_{i=1..n_M}$ are defined prior to any discretization. A classical choice consists in using what is called an extraction projector of the "linear part" of the fields. Figure 5.15 illustrates this basis of affine functions in two dimensions. In this case, components $[F^M]_i$ correspond to the resultants, moments, and extension of the interface. The same basis is used for $\mathcal{W}^M_{EE'}$. Then, components $[W^M]_i$ correspond to the translations, rotations, and stretching of the interface.

In order to transfer the "global" data to all of structure Ω, the macro loads are required to verify the transmission conditions systematically. The associated space is denoted by $\mathcal{F}^M_{ad}$.

$$\mathcal{F}^M_{ad} = \{\underline{F}^M \,|\, \forall E, \forall E' \in \mathbf{V}_E, \underline{F}^M_E + \underline{F}^M_{E'} = 0\} \qquad [5.8]$$

where $\mathbf{V}_E$ represents the set of substructures which are neighbors of Ω_E.

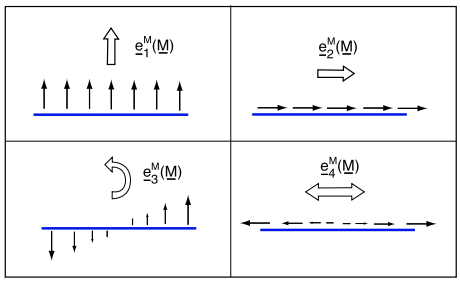


Figure 5.15. *The linear macroscopic basis ($n_M = 4$).*

5.7.1.3. *The iterative resolution strategy*

The solution **s** of the problem is determined using the LATIN iterative algorithm developed by Ladevèze [LAD 99]. Unlike most "classical" methods, in which calculation of the displacement/load response requires an incremental procedure and an iteration at each time step, the LATIN method provides a complete admissible response (over the whole time interval) at each iteration, and consists in adjusting this response until the solution is reached. Figure 5.16 gives a schematic illustration of the behavior of the LATIN resolution method compared to that of an incremental iterative resolution strategy.

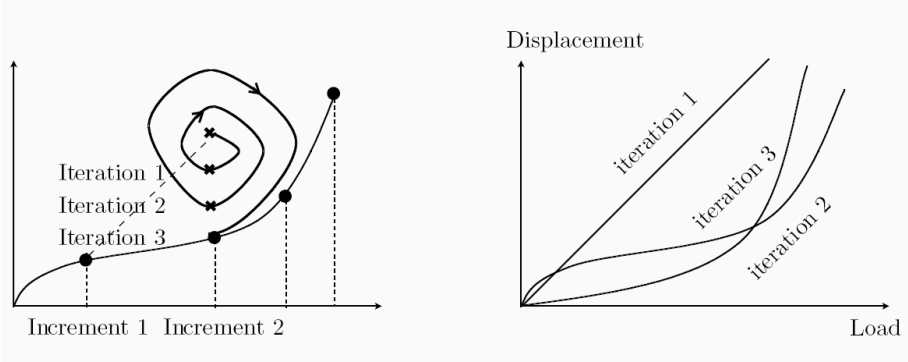


Figure 5.16. *Schematic representation of the behavior of two resolution strategies.*

The basic principle of the LATIN method consists in separating the difficulties by resolving successively two groups of equations A_d and Γ defined by:

$$A_d \left\| \begin{array}{ll} - & \text{the static admissibility of } (\sigma_E, \underline{F}_E) \\ - & \text{the kinematic admissibility of } (\epsilon_E, \underline{W}_E) \\ - & \text{the admissibility of } \bigcup_{E\in\mathbf{E}}\{\underline{F}_E^M\} \in \mathcal{F}_{ad}^M \end{array} \right.$$

$$\Gamma \left\| \begin{array}{ll} - & \text{the behavior of the interfaces} \end{array} \right.$$

Γ includes the local equations, which may be non-linear, and A_d includes the linear equations, which may be global.

The strategy consists in seeking a solution which verifies the equations of A_d and the equations of Γ alternatively, using two search directions E^+ and E^- (Figure 5.17).

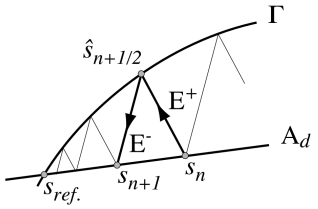


Figure 5.17. *Schematic representation of an iteration of the LATIN method.*

In the local stage, given $\mathbf{s}_n \in A_d$, we seek $\hat{\mathbf{s}}_{n+1/2} \in \Gamma$ which verifies:

$$(\hat{\underline{F}}_E - \underline{F}_E) - k^+(\hat{\underline{W}}_E - \underline{W}_E) = 0 \quad \forall M \in \Gamma_{EE'} \qquad [5.9]$$

k^+ being a positive scalar associated with the search direction E^+.

In the linear stage, given $\hat{\mathbf{s}}_{n+1/2} \in \Gamma$, we seek $\mathbf{s}_{n+1} \in A_d$ which verifies:

$$(\underline{F}_E - \hat{\underline{F}}_E) + k^-(\underline{W}_E - \hat{\underline{W}}_E) = 0 \quad \forall M \in \Gamma_{EE'} \qquad [5.10]$$

Generally, we take $k^- = k^+ = k$.

In the case of a multiscale resolution, taking into account the macro admissibility, $\bigcup_{E\in\mathbf{E}}\{\underline{F}_E^M\} \in \mathcal{F}_{ad}^M$ changes the search direction E^-:

$$(\underline{F}_E - \hat{\underline{F}}_E) + k^-(\underline{W}_E - \hat{\underline{W}}_E - \widetilde{\underline{W}}_E^M) = 0 \quad \forall M \in \Gamma_{EE'} \qquad [5.11]$$

where $\underline{\widetilde{W}}_E^M$ is the Lagrange multiplier which corresponds to the admissibility constraint of the macro loads and $\underline{\widetilde{W}}_E^M \in \mathcal{W}_{ad,0}^M$ is the space of the macro displacements which are continuous at the interfaces and zero over $\partial_1\Omega$ [LAD 03].

5.7.2. *Behavior of the interfaces*

The behavior of the interface $\Gamma_{EE'}$ between two substructures Ω_E and $\Omega_{E'}$ depends on the connection that the interface is intended to represent. This behavior can be expressed as a mixed constitutive relation between the interdisplacements and the interloads acting over the interface. Let us give two examples of interface behavior:

Perfect interface
The displacements are continuous across the interface and the loads are in equilibrium. Then, the constitutive relations lead to the two equations: $\underline{F}_E + \underline{F}_{E'} = 0$ and $\underline{W}_E - \underline{W}_{E'} = 0$.

Contact interface with friction and a gap In order to verify the friction conditions optimally, it is necessary to take into account their evolution throughout the loading [CHA 97]. Thus, the Coulomb friction problem is discretized in time. μ denotes Coulomb's friction coefficient, $\underline{n}$ represents the outward normal to $\Gamma_{EE'}$ with respect to Ω_E at the current point, and j is the initial gap (Figure 5.18).

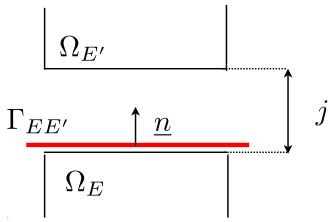


Figure 5.18. *The notations for a contact interface.*

$\mathbf{P}_t$ designates the orthogonal projector associated with interface $\Gamma_{EE'}$. Then, the constitutive relation comes down to:

Unilateral contact

– Delamination

$$\text{If } \underline{n} \cdot (\Delta \underline{W}_{E'}^t - \Delta \underline{W}_E^t) > j - \underline{n} \cdot (\underline{W}_{E'}^{t-1} - \underline{W}_E^{t-1}) \text{ then } \underline{F}_E^t = \underline{F}_{E'}^t = 0$$

or $\Delta \underline{W}_E^t = \underline{W}_E^t - \underline{W}_E^{t-1}$

– Contact

$$\text{If } \underline{n}\cdot(\Delta\underline{W}^t_{E'}-\Delta\underline{W}^t_E)=j-\underline{n}\cdot(\underline{W}^{t-1}_{E'}-\underline{W}^{t-1}_E) \text{ then } \left\{\begin{array}{rcl}\underline{F}^t_E+\underline{F}^t_{E'} & = & 0\\ \underline{n}\cdot\underline{F}^t_E & < & 0\end{array}\right.$$

Friction conditions

– Adhesion

$$\text{If } \|\mathbf{P}_t\underline{F}^t_E\|<\mu|\underline{n}\cdot\underline{F}^t_E| \text{ then } \mathbf{P}_t(\Delta\underline{W}^t_{E'}-\Delta\underline{W}^t_E)=0$$

– Slipping

$$\text{If } \|\mathbf{P}_t\underline{F}^t_E\|=\mu|\underline{n}\cdot\underline{F}^t_E| \text{ then } \left\{\begin{array}{rcl}\mathbf{P}_t(\Delta\underline{W}^t_{E'}-\Delta\underline{W}^t_E)\wedge\mathbf{P}_t\underline{F}^t_E & = & 0\\ \mathbf{P}_t(\Delta\underline{W}^t_{E'}-\Delta\underline{W}^t_E)\cdot\mathbf{P}_t\underline{F}^t_E & \geq & 0\end{array}\right.$$

5.7.3. *Resolution*

In this section, we briefly describe the different calculation steps involved in the course of a LATIN iteration.

5.7.3.1. *Resolution in the linear stage: determination of* $\mathbf{s}_{n+1} \in A_d$

5.7.3.1.1. The micro problem

For a substructure Ω_E, the weak formulation in displacement associated with equations [5.1] to [5.3] and the verification of the search direction [5.11] lead to the following micro problem:

Find $(\underline{u}_E, \underline{W}_E)$ which verifies:

$$\int_{\Omega_E} Tr(\epsilon(\underline{u}_E):\mathbf{D}:\epsilon(\underline{u}^*))d\Omega+\int_{\partial\Omega_E} k\underline{W}_E\cdot\underline{W}^*d\Gamma=$$

$$\int_{\partial\Omega_E}(\hat{\underline{F}}_E+k\hat{\underline{W}}_E+k\widetilde{\underline{W}}^M_E)\cdot\underline{W}^*d\Gamma \qquad [5.12]$$

A standard finite element discretization is used to interpolate the interface quantities as well as the displacement fields in the substructures. Finally, after discretizing equation [5.12], we get:

$$([K_E] + [k_E])[\underline{u}_E] = [\widehat{\underline{F}}_E] + k[\widetilde{\underline{W}}_E^M] \qquad [5.13]$$

where $[K_E]$ is the "classical" finite element stiffness matrix and $[k_E]$ is the stiffness matrix of the interface. $\widetilde{\underline{W}}_E^M$ is the only macro unknown of problem [5.12]. Its determination requires the resolution of a macro problem over the whole set of substructures.

5.7.3.1.2. The macro problem

Because the micro problem is linear over Ω_E, using [5.11], we can define $\mathbf{L}_E^F$, a homogenized operator relating the macro loads to the Lagrange multipliers:

$$\underline{F}_E^M = \mathbf{L}_E^F(\widetilde{\underline{W}}_E^M) + \hat{\underline{F}}_{E,d}^M$$

where $\hat{\underline{F}}_{E,d}^M$ is the "given" macro loading. Then, the macro problem over the whole set of substructures becomes:

Find $\widetilde{\underline{W}}_E^M$ such that:

$$\sum_E \int_{\partial\Omega_E} \widetilde{\underline{W}}_E^{M*} \cdot \left(\mathbf{L}_E^F(\widetilde{\underline{W}}_E^M) + \hat{\underline{F}}_{E,d}^M\right) d\Gamma = \sum_E \int_{\partial\Omega_E \cap \partial_2\Omega} \widetilde{\underline{W}}_E^{M*} \cdot \underline{F}_d \, d\Gamma \qquad [5.14]$$

After discretization, we get:

$$[\mathbf{L}^F]_{\mathbf{e}^M}[\widetilde{\underline{W}}^M]_{\mathbf{e}^M} = [\widehat{\underline{F}}_d^M]_{\mathbf{e}^M} + [\underline{F}_d^M]_{\mathbf{e}^M} \qquad [5.15]$$

$[.]_{\mathbf{e}^M}$ designates the set of components of the quantity being considered in the macro basis. Therefore, the size of the macro problem is $n_M \times n_i$, where n_i is the number of interfaces and n_M the dimension of the macro basis of space $\mathcal{F}_{EE'}^M$ (Figure 5.15).

5.7.3.2. *Resolution in the local stage: determination of* $\hat{\mathbf{s}}_{n+1/2} \in \Gamma$

The state of each point of the contact interface is given explicitly by two indicators, g_N and $\underline{g}_T$ [CHA 99, LAD 02]. In the case of contact with friction, introducing a time discretization, these two quantities are deduced from the solution of the previous step.

The normal contact indicator g_N is defined as:

$$g_N^{t+1} = \frac{1}{2}\underline{n}.(\underline{W}_{E'}^{t+1} - \underline{W}_{E}^{t+1} + \Delta\hat{\underline{W}}_{E'}^{t} - \Delta\hat{\underline{W}}_{E}^{t}) - \frac{j}{2} - \frac{1}{2k}\underline{n}.(\underline{F}_{E'}^{t+1} - \underline{F}_{E}^{t+1}) \qquad [5.16]$$

The friction indicator $\underline{g}_T$ is defined as:

$$\underline{g}_T^{t+1} = \frac{k}{2}\mathbf{P}_t(\Delta\hat{\underline{W}}_{E'}^{t} - \Delta\hat{\underline{W}}_{E}^{t}) - \frac{1}{2}\mathbf{P}_t(\underline{F}_{E'}^{t+1} - \underline{F}_{E}^{t+1}) \qquad [5.17]$$

For normal contact: if $g_N^{t+1} > 0$, delamination occurs; if $g_N^{t+1} \leq 0$, the interfaces are in contact. With respect to tangential contact: if $\underline{g}_T^{t+1} < \mu|\underline{n} \cdot \hat{\underline{F}}_E^t|$, slipping occurs; if $\underline{g}_T^{t+1} \geq \mu|\underline{n} \cdot \hat{\underline{F}}_E^t|$, there is adhesion of the contact surfaces.

Finally, depending on the contact case, the normal and tangential components of the interface quantities $\hat{\underline{W}}_E^{t+1}$, $\hat{\underline{W}}_{E'}^{t+1}$, $\hat{\underline{F}}_E^{t+1}$, and $\hat{\underline{F}}_{E'}^{t+1}$ are calculated.

5.7.3.3. *Stopping criterion*

The non-incremental character of the LATIN strategy gives very easy access to a convergence indicator. Indeed, any criterion based on a measure of the distance between two consecutive solutions of the same iteration $\|\mathbf{s}_{n+1} - \hat{\mathbf{s}}_{n+1/2}\|$ is a relevant error indicator [LAD 99]. This indicator is calculated at the end of each iteration, and when its value becomes less than a threshold ϵ_l, the algorithm is stopped.

5.7.4. *The resolution algorithm*

In summary, Figure 5.19 gives the general resolution algorithm. For each step, quantities *before* the arrow denote the quantities known *a priori* and quantities *after* the arrow denote the quantities which are calculated.

5.7.5. *Relevance of the multiscale approach in the case of multilevel optimization*

As mentioned previously, the multiscale approach is based on a decomposition of the fields at the interfaces in micro quantities and macro quantities. Thus, in the resolution process, the macro quantities are obtained through a homogenized problem defined over the whole set of the substructures.

This homogenized problem constitutes a first modeling level, defined intrinsically within the method. If we consider the 2D problem of a beam in bending, the macroscopic quantities can be viewed as the resultants and moments of the internal loads

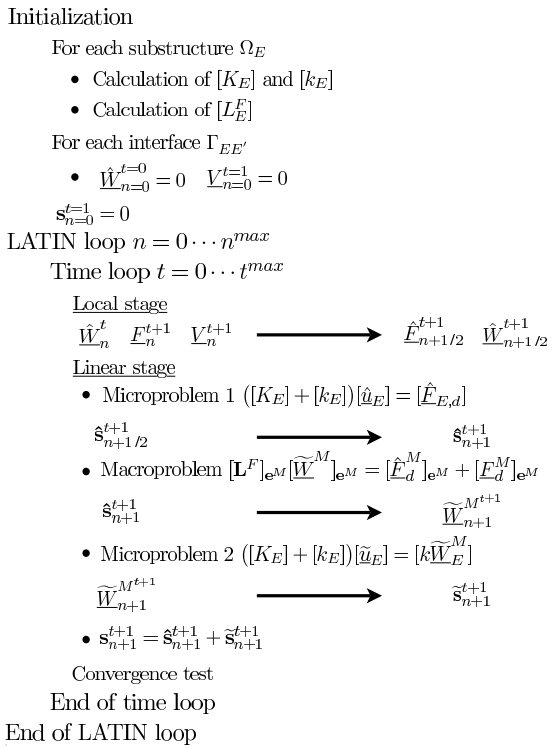


Figure 5.19. *The resolution algorithm.*

at each point where there is an interface: in this case, the model associated with the resolution of the macroscopic problem is a beam-type model such as can be found in strength of materials.

Thus, the micro quantities, determined as corrections to this first modeling level, constitute a second modeling level.

In other words, the multiscale strategy proposed introduces two modeling levels naturally, and these are treated simultaneously within the resolution algorithm. Thus, when the optimizer invokes the strategy, two levels of model are generated transparently. Furthermore, as will be seen in the next section, the solutions associated with a set of parameters of these two modeling levels are preserved when a new calculation associated with another set of parameters is performed.

5.7.6. *The multiresolution strategy*

Since optimization generates a large number of calculations, it is crucial to introduce an appropriate resolution strategy which enables a reduction in computing time when the parameters change. Different strategies, called *mutiresolution strategies*, have been presented in the context of topological optimization, for example in [KIM 00]. The multiresolution method used here was developed by [BOU 03, BOU 04]. It is based on the fact that the LATIN algorithm can be initialized using any solution verifying the admissibility conditions ($\mathbf{s} \in A_d$). In the case of a parametric study, for a given set of parameters, the LATIN loop is reinitialized with the converged solution (which necessarily belongs to A_d) corresponding to another set of parameters. When a parameter evolves only slightly, the global solution of the problem also changes only slightly. Thus, using multiresolution strategy, convergence is achieved more rapidly in a smaller number of iterations.

The multiscale strategy coupled with multiresolution was validated in [BOU 07].

Description of the structural detail

Taking into account the shape and/or the position of structural details constitutes an important stage in the dimensioning of a mechanical system. The multiscale aspect we have introduced provides a means of assessing the global behavior of the structure (on the macro scale) as well as including the influence of the structural detail in a simple way (on the micro scale). The optimization of such details involves prohibitive computing times if a complete optimization of the refined model is attempted and, therefore, it requires the application of a multilevel (multiscale) strategy. In this case, the description of a structural detail is carried out independently of the finite element mesh through the use of the X-FEM. The level set technique enables us to modify its shape and/or its position in the course of a calculation very easily and inexpensively.

5.7.7. *The X-FEM method*

In order to avoid the difficulties of meshing at the micro level, an enrichment technique such as X-FEM is used for the representation of the structural detail [GUI 06]. Since the multiscale aspect is introduced only at the interface level, it is possible to enrich the displacement field $\underline{u}_E$ within a substructure E using the technique illustrated in [MOË 99, STO 00]. In the case of a circular hole:

– The X-FEM interpolation of the displacement field in substructure E is given by [SUK 01]:

$$\underline{u}_{E_h}(\underline{x}) = \sum_{i \in N} \varphi_i(\underline{x})\, H(\underline{x})\, \underline{u}_i \text{ where } H(\underline{x}) = \begin{cases} 1 & \text{if } \underline{x} \text{ is in the material} \\ 0 & \text{if } \underline{x} \text{ is in the void} \end{cases} \qquad [5.18]$$

where N, $\underline{u}_i$, and H denote respectively the set of the nodes of the mesh, the corresponding degrees of freedom, and the enrichment function. Only the elements crossed by the boundary of the hole require special treatment.

– The numerical integration of the weak formulation is not performed on the portion of the element located in the void.

– The nodes inside the hole which are not connected to any element crossed by the hole's boundary are eliminated from the mesh by removing their degrees of freedom from the final system.

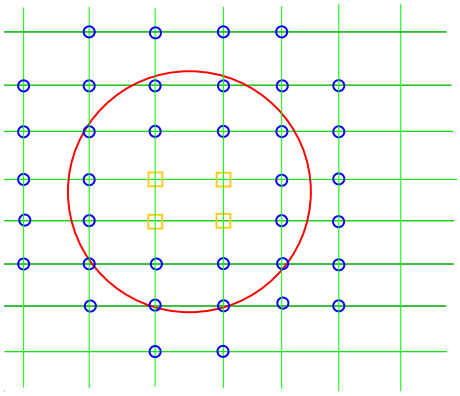


Figure 5.20. *A hole positioned over a mesh. The circle nodes require a specific treatment, while the nodes marked with a square are eliminated.*

5.7.8. *The level set method*

The level set function method proposed by [OSH 88] is a numerical technique designed to follow the evolution of structural details easily, since it is independent of the finite element mesh. In the case of a hole, the level set function (which is the distance function) is given by:

$$\varphi(\underline{x}) = min(\|\underline{x} - \underline{x}_c\| - r_c) \qquad [5.19]$$

where $\underline{x}_c$ and r_c denote the positions of the origin and the radius of the hole, respectively. Then, the enrichment function H can be easily expressed as a function of the distance function φ as follows:

$$H(\underline{x}) = \begin{cases} 1 & \text{if} \quad \varphi(\underline{x}) \geq 0 \\ 0 & \text{if} \quad \varphi(\underline{x}) < 0 \end{cases}$$

So far, we have presented strategies in the domain of structural analysis (for which numerical examples are proposed in section 5.9). Now, let us address some problems in the domain of aerodynamics.

5.8. A multilevel method for aerodynamics using an inexact pre-evaluation approach

Modern semi-stochastic optimization methods like genetic algorithms [MIC 92] and particle swarm optimization [VEN 03] have been found to be capable of solving practical optimization problems. Among their many advantages are their ability to handle non-smooth functions (since gradient information is not required), and the possibility of finding global optimal solutions. A distinguishing feature of these methods is that they operate with a *population/swarm*, i.e. they make use of multiple candidate solutions at each step of their iteration. This requires computation of the *cost/fitness* function for each candidate in every optimization iteration. The ability to locate the global optimum depends on sufficient exploration of the design space, which requires use of a sufficiently large population size. This is especially true when the cost function is multimodal and the dimension of the design variable space is high. With the increasing use of high-fidelity models, e.g. Navier–Stokes equations for flow analysis, the computation of the cost function for a single design can be costly in terms of time and resource use. The combination of such high-fidelity analysis tools with population-based optimization techniques can render them impractical or severely limit the size of the population that can be used.

To overcome this barrier, several researchers have used *surrogate models* or *metamodels* [BUC 05, GIA 02, EMM 06, JIN 05, ONG 04] in place of the costly evaluation tool. These surrogate models are inexpensive compared to the exact model. There are several ways in which a surrogate model can be developed:

– Data-fitting models: an approximation to the cost function is constructed using the available data. These data may be either generated specifically for constructing the model or may be taken from the initial few iterations of the optimization method. Examples of data-fitting models are polynomials (usually quadratic, also known as response surface models) [JIN 05], artificial neural networks (such as multilayer perceptrons, radial basis function networks) [JIN 05, EMM 06], and Gaussian process models (kriging) [BUC 05]. These models can be either global, making use of all available data, or local, making use of only a small set of data around the point where the function is to be approximated. Global models have been used as a complete replacement of the original cost function, with optimization being carried out on the surrogate model. Local models have typically been used as preconditioners to accelerate the exploration of the search space. These types of model are used in Chapter 3.

– Variable convergence models: the cost function usually depends on the numerical solution of a PDE. Most numerical methods are iterative in nature and contain a

stopping criterion which is measured in terms of a solution residual. To get an accurate solution, a small value of the residual is usually used. Such an accurate solution may be unnecessary when all we want is an estimate of a cost function, which is usually some integral that converges much faster. In such a situation, the stopping criterion can be relaxed, thereby considerably reducing the time taken for a single computation.

– Variable resolution models: in these models, a hierarchy of grids is used and the surrogate model is simply the costly evaluation tool run on a coarse grid.

– Variable fidelity models: in these models, a hierarchy of physical models is used, for example Euler equations (surrogate model) and RANS equations (exact model). Even when a high-fidelity model like RANS is used, we can use a wall function approximation as a surrogate model and a turbulence model applied up to to the wall as the exact model.

In the following sections, we consider data-fitting models, particularly radial basis functions and Gaussian random process models, also known as kriging. Both these methods have been found to be effective in interpolation of high-dimensional data with small numbers of data points as compared to polynomial-based methods. Data fitting models have been extensively used for optimization of costly functions [JIN 05]. Quadratic models were frequently used in the past but their lack of accuracy has led to the development of more sophisticated approximation methods like neural networks, radial basis functions, and kriging. There are several variations in the use of metamodels for optimization. In off-line trained methods, a metamodel is first constructed by generating a set of data points in the design space and evaluating the cost function at these points. This metamodel is then used to optimize the cost function without recourse to the exact function. The success of this method relies on the ability to construct an accurate metamodel, which is doubtful for realistic problems, which usually involve large numbers of design variables and complex function landscapes. On-line trained methods construct and update the metamodel as and when required and are closely integrated into the optimization loop. Whenever a new function value is available, the metamodel is updated and the optimization proceeds using the new metamodel. The metamodel becomes progressively more accurate as more and more data points are included in its construction.

Giannakoglou [GIA 02] has proposed a two-level evaluation strategy, called inexact pre-evaluation (IPE), to reduce the computation time related to GAs. It relies on the observation that numerous cost function evaluations are useless, since numerous individuals do not survive to the selection operator. Hence, it is not necessary to determine their fitness accurately. The strategy proposed by Giannakoglou consists in using metamodels to pre-evaluate the fitness of the individuals in the population. Then only a small portion of the population, corresponding to the most promising individuals, is accurately evaluated using the original and expensive model.

Inspired by the success of GAs combined with metamodels and IPE, we study the application of a similar strategy to particle swarm optimization. PSO was introduced in [KEN 95] as a simplified social model. It mimics the behavior of bird flocking and is based on rules that enable sudden direction changes, scattering, regrouping, etc. These moves are motivated in nature by food seeking or predator avoiding, and can be implemented in a simple algorithm for global optimization. PSO also requires a large number of function evaluations since it requires a large number of particles to locate the optimum effectively. Hence we propose a metamodel-assisted PSO in which *local RBF approximations are used to pre-evaluate the particles*. Then a small percentage of particles are selected (prescreening) for exact evaluations. We also propose a new prescreening criterion which is specific to PSO and automatically determines the number of exact evaluations. The proposed algorithm is applied to the aerodynamic shape optimization of a supersonic business jet and a transonic wing. In both cases, a substantial reduction in the number of CFD evaluations is achieved, while finding optimal shapes that are as good as in the case of CFD evaluations alone.

5.8.1. *Particle swarm optimization*

PSO is modeled on the behavior of a swarm of animals when they hunt for food or avoid predators [MIL 07]. In nature, a swarm of animals is found to exhibit very complex behavior and to be capable of solving difficult problems like finding the shortest distance to a food source. However, the rules that govern the behavior of each animal are thought to be simple. Animals are known to communicate the information they have discovered to their neighbors and then act upon that individually. The individuals cooperate through self-organization but without any central control. The interaction of a large number of animals acting independently according to some simple rules produces highly organized structures and behaviors.

In PSO, a swarm of particles wanders around in the design space according to some specified velocity. The position of each particle corresponds to one set of design variables and it has an associated value of the cost function. Each particle remembers the best position it has discovered in its entire lifetime (local memory) and also knows the best position discovered by its neighbors and the whole swarm (global memory). The velocity of each particle is such as to pull it towards its own memory and that of the swarm. While there are many variants of the PSO algorithm, the one we use is described below and complete details are available in [DUV 06]. The algorithm is given for a function minimization problem

$$\min_{x_l \leq x \leq x_u} J(x)$$

Algorithm: *Particle swarm optimization*

1) Set $n = 0$

2) Randomly initialize the positions and velocities $\{x_k^n, v_k^n\}, k = 1, \ldots, K$.

3) Compute cost function values $J(x_k^n), k = 1, \ldots, K$.

4) Update the local and global memory

$$x_{*,k}^n = argmin_{0 \le s \le n} J(x_k^s), \quad x_*^n = argmin_{0 \le s \le n, 1 \le k \le K} J(x_k^s) \qquad [5.20]$$

5) Update the particle velocities

$$v_k^{n+1} = \omega^n v_k^n + c_1 r_{1,k}^n (x_{*,k}^n - x_k^n) + c_2 r_{2,k}^n (x_*^n - x_k^n) \qquad [5.21]$$

6) Apply craziness operator to the velocities.

7) Update the position of the particles

$$x_k^{n+1} = x_k^n + v_k^{n+1} \qquad [5.22]$$

8) Limit new particle positions to lie within $[x_l, x_u]$ using reflection at the boundaries.

9) If $n < N_{\max}$, then $n = n + 1$ and go to step (iii), else STOP.

Apart from the above basic algorithm, we use several additional strategies to enhance the efficiency of PSO. The inertia parameter ω is decreased during the iterations as proposed by Fourie and Groenwold [FOU 02]. A starting ω^o is chosen with a large value in order to promote an exploratory search. Its value is then decreased by a factor α if no improved solution is found within h consecutive time steps:

$$\text{If } J(x_*^n) = J(x_*^{n-h}) \text{ then } \omega^n = \alpha \omega^{n-1}$$

with $\alpha \in (0, 1)$. Therefore, if the exploratory search fails, convergence towards the best locations ever found is promoted. A craziness operator is implemented on the velocity [FOU 02], which is inspired by the mutation operator in GAs. A craziness probability $p_c \in [0, 1]$ is chosen; then, at each time step and for each particle, the velocity direction is randomly modified with the probability p_c, but the velocity modulus is kept constant. Large random perturbations therefore occur at the beginning of the optimization procedure, promoting random global search, whereas small random perturbations are performed when the swarm is close to the solution, promoting random local search. This approach is inspired by the so-called non-uniform mutation operator in GAs [MIC 92]. Finally, an upper limit on velocity, as recommended by Shi and Eberhart [SHI 98], in order to improve the stability and convergence rate of PSO is also used.

In the original algorithm proposed by Kennedy and Eberhart [KEN 95], the random numbers r_1, r_2 are scalars, i.e. one random number is used for all the velocity

components of a particle. In practical implementations, researchers have used both a scalar and vector version of random numbers. In the vector version, a different random number is used for each component of the velocity vector. This is equivalent to using random diagonal matrices for r_1 and r_2. In [WIL 05], the author has investigated the difference in PSO performance between these versions and concludes that the scalar version is susceptible to becoming trapped in a line search, while the vector version does not have this problem. The vector version is also preferred for use with metamodels since it has space-filling characteristics.

5.8.2. *Metamodel assisted PSO with inexact pre-evaluation (IPE)*

Like genetic algorithms, PSO is a rank-based algorithm; the actual magnitude of the cost function of each particle is not important, only their relative ordering matters. An examination of the PSO algorithm shows that the main driving factors are the local and global memories. Most of the cost functions are discarded except when they improve the local memory of the particle. Hence in the PSO context, an inexact pre-evaluation strategy seems to be advantageous in identifying promising particles, i.e. particles whose local memory is expected to improve, which can then be evaluated on the exact function. When updating the local and global memories, the cost functions are of mixed type; some particles have cost functions evaluated on the metamodel and a few are evaluated using the exact model. If the memories are updated using cost functions evaluated on the metamodel, there is the possibility that the memory may improve due to errors in the cost functions. This erroneous memory may cause PSO to converge to it or may lead to wasteful searches. The memories are therefore updated using only the exactly evaluated cost functions. We propose a metamodel-assisted PSO with inexact pre-evaluation as follows; the first N_e iterations of PSO are performed with exact function evaluations which are stored in a database. In the present work $N_e = 10$ is used. In the subsequent iterations the metamodel is used to prescreen the particles and only a small percentage of particles are evaluated on the exact function.

Algorithm: *Particle swarm optimization with IPE*

1) Set $n = 0$.

2) Randomly initialize the positions and velocities $\{x_k^n, v_k^n\}, k = 1, \ldots, K$.

3) If $n \leq N_e$, compute cost function associated with the particle positions $J(x_k^n), k = 1, \ldots, K$ using the exact model, else compute the cost function using metamodel $\hat{J}(x_k^n), k = 1, \ldots, K$.

4) If $n > N_e$, select a subset of particles S^n based on a prescreening criterion and evaluate the exact cost function for these particles. Store the exact cost functions in the database.

5) Update the local and global memories *using only the exactly evaluated cost functions*

$$x^n_{*,k} = argmin_{0\le s\le n} J(x^s_k), \quad x^n_* = argmin_{0\le s\le n, 1\le k\le K} J(x^s_k) \qquad [5.23]$$

6) Store exactly evaluated function values in a database.

7) Update the particle velocities

$$v^{n+1}_k = \omega^n v^n_k + c_1 r^n_{1,k}(x^n_{*,k} - x^n_k) + c_2 r^n_{2,k}(x^n_* - x^n_k) \qquad [5.24]$$

8) Apply a craziness operator to the velocities.

9) Update the position of the particles

$$x^{n+1}_k = x^n_k + v^{n+1}_k \qquad [5.25]$$

10) Limit new particle positions to lie within $[x_l, x_u]$ using reflection at the boundaries.

11) If $n < N_{\max}$, then $n = n + 1$ and go to step (iii), else STOP.

The important aspect of metamodel-assisted optimization is the criterion used to select the set S of particles whose function value will be exactly evaluated. Giannakoglou [EMM 06] discusses several prescreening criteria based on the estimated fitness function and variance of the estimation whenever available, as in the case of Gaussian random process models. The prescreening criteria are based on the notion of *improvement*. Let $J_{\min}$ be the current minimum function value and $\hat{J}(x)$ be the function value predicted by the metamodel for a new design point x. We can define an index of improvement for the design x as

$$I(x) = \begin{cases} 0 & \text{if } \hat{J}(x) > J_{\min} \\ J_{\min} - \hat{J}(x) & \text{otherwise} \end{cases} \qquad [5.26]$$

Designs with larger value of this index are likely to lead to a reduction in the cost function and should be evaluated on the exact function. Some metamodels like kriging also give an estimate of the error in the approximation. This information can be useful for exploring those regions of the design space which are not sufficiently probed. We do not consider these other criteria; see [EMM 06] for further details.

In the present work we use interpolating RBF metamodels, which do not provide an estimate of the variance. The prescreening is therefore based only on the estimated cost function value and we investigate two different criteria:

– After the IPE phase, the particles are sorted in order of increasing cost function and a specified percentage of the *best* particles, i.e. those with small cost function values, are selected for exact evaluation.

– We also propose a new prescreening criterion for PSO as follows: the set S^n consists of all particles whose local memory value is predicted to be reduced in the IPE phase, i.e.

$$S^n = \{k : \hat{J}(x_k^n) < J(x_{*,k}^{n-1})\} \quad [5.27]$$

The second criterion is similar to the index of improvement but the minimum function value is that of the individual particle memory. All particles whose index is positive (non-zero) are evaluated on the exact function. Note that we do not specify the percentage of particles that are exactly evaluated; *the number of exact function evaluations is automatically determined* and we expect this number to adapt itself as the cost function is progressively reduced. Note that in this PSO + IPE approach, both *the local and global memories always consist of exactly evaluated particles*.

5.9. Numerical examples

In this last part, we will present:

– Four examples of applications of multilevel optimization of a structural detail. The optimization algorithms used in this study are descent algorithms in which the gradients of the cost function are calculated through finite differences.

– Two examples of multilevel optimization in aerodynamics, focusing more particularly on the shape optimization of the wing of a supersonic business jet.

5.9.1. *Example 1: optimization of the position of a hole*

Let us consider a structure Ω_1, fixed along its length and in contact with friction with a structure Ω_2 subjected to a uniform pressure $p = 100\ MPa$ over its edges. The material's constants and the friction coefficient are $E = 210000\ MPa$, $\nu = 0.3$, and $\mu = 0.6$. The domain was divided into 13 identical substructures and the contact interface is represented in bold in Figure 5.21. Each substructure was divided into 18×18 quadrangular elements.

A hole with an 8 *mm* diameter is cut in the upper structure. The coordinates of its center, x_c and y_c, are the design variables of the problem. They must be optimized so the normal pressure distribution at the contact interface is as uniform as possible. Then, the cost function f is given by:

$$f = \frac{(p_{max} - p_{min})}{p_{avg}}$$

where p_{max}, p_{min}, and p_{avg} are the maximum, minimum, and average pressures, respectively.

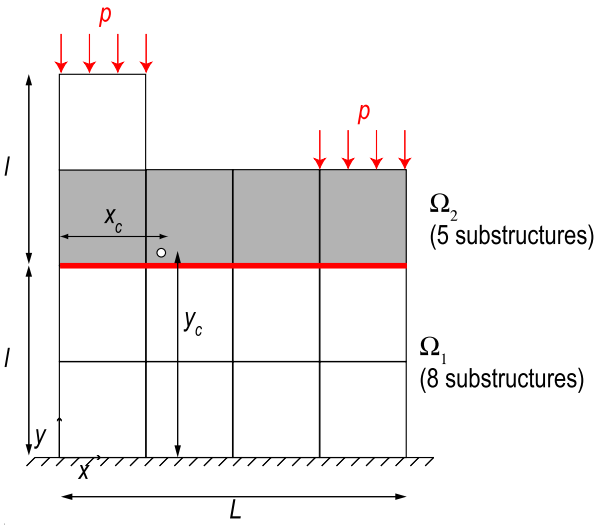


Figure 5.21. *The model,*

The optimization was carried out in three steps according to the scheme shown in Figure 5.11.

First, we constructed a metamodel representing the first modeling level. In order to do this, we performed a multiresolution calculation of the cost function for different positions of the hole taken inside the grey zone (the black points in Figure 5.22). Then, we obtained the response surface by cubic interpolation.

Using this metamodel, the inexpensive optimization, using a genetic algorithm led to the zone(s) of interest very rapidly. In practice, the information obtained at the end of the optimization of the cost function was the substructure within which the hole must lie.

We then proceeded with the refined optimization itself. The cost function was calculated at each call using the multiscale calculation code. The minimization was performed under the constraint that the hole must remain inside the previously identified substructure. Taking $(x_c^0, y_c^0) = (83, 183.5)$ as the initial values, the algorithm converged to the optimum position $(x_c^*, y_c^*) = (79.5, 186, 9)$ after 14 optimization loops and 84 calls to the cost function f.

With our approach, only the first call to f during the construction of the metamodel or during the optimization process was really expensive. For subsequent calculations, when the position of the hole varied, we initialized the LATIN algorithm with

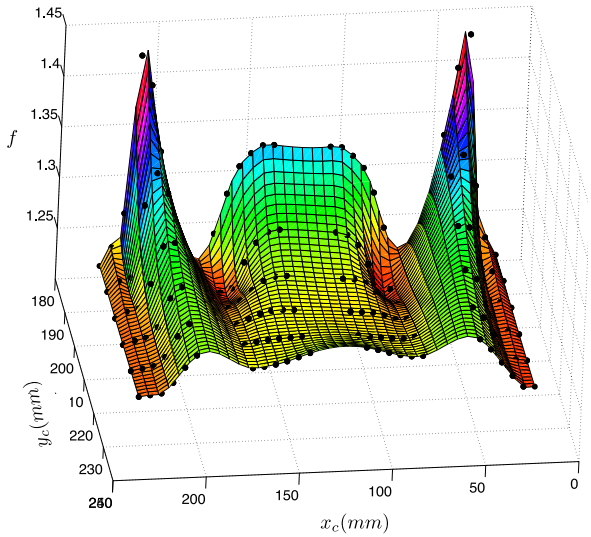


Figure 5.22. *The response surface.*

the converged solution from the previous calculation. Furthermore, thanks to the domain decomposition, we recalculated only the stiffness matrices of the substructures concerned with the detail. To illustrate thist point, Table 5.1 compares, for the two calculation phases involving the simulator, the CPU times required by our method with those using a "classical" method (without multiresolution). The ratio "classical time/multiresolution time", which is greater than 6, demonstrates the power of the multiscale/multiresolution method. During the optimization phase of the refined model, two types of calls to the cost function must be distinguished: "actual" calls, 45 in this case, which required an average of 50 s CPU time, and all the calls required to evaluate the gradients. The latter involved smaller variations of the position of the hole and, therefore, generated lower computation times, in the order of 20 s.

5.9.2. *Example 2: optimization of the geometry of contact surfaces*

This 2D example was inspired by [LI 05]. Here, the objective is to optimize the shape of the contact surface between two structures in order to demonstrate the efficiency of the multiresolution strategy when the interface parameters vary.

Let us consider structure Ω_1, divided into four substructures, in frictionless contact ($\mu = 0$) with structure Ω_2, itself divided into six substructures (Figure 5.23). A pressure $P = 50$ MPa is applied in a single time step. For a perfectly planar contact surface Γ_c (with no gaps), a stress concentration occurs at the edges. The objective

Step	Number of calls to the cost function	LATIN computing time	LATIN computing time (multiresolution)
Construction of the metamodel (LATIN calculation)	144	144×326 s	$326 + 10,716$ s
Optimization of the refined model (descent algorithm + LATIN)	84	84×386 s	$386 + 2,595$ s
	total	79,368 s	14,023 s

Table 5.1. *Gain due to the multiresolution strategy in the multilevel model optimization strategy.*

is therefore to optimize the profile of the contact surface in order to obtain the most regular pressure distribution possible (thus reducing the edge effects). Figure 5.23 represents a zoomed view of the geometry being considered.

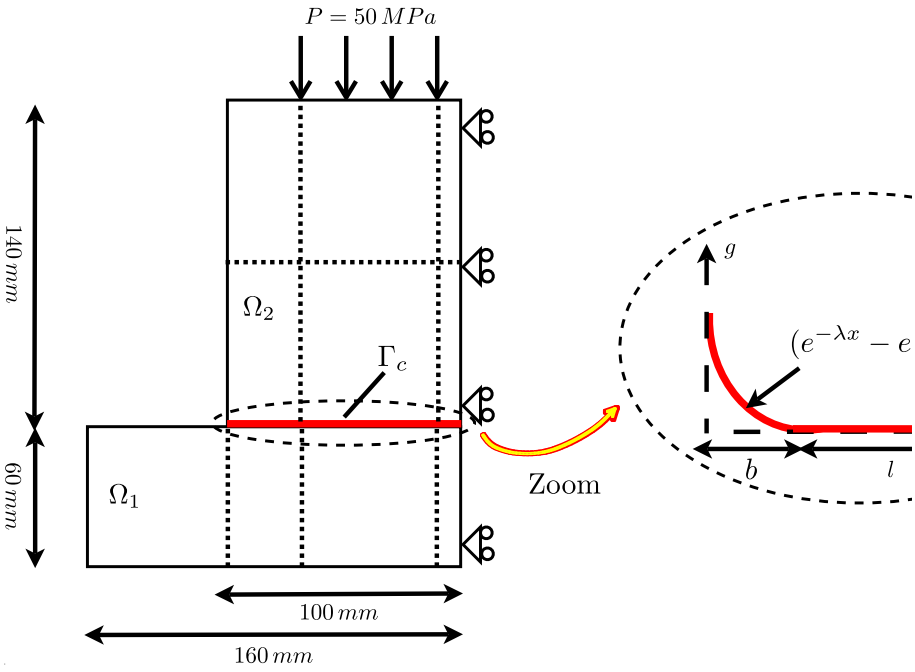


Figure 5.23. *The model, with zoom on the geometry of the contact surface.*

The design parameters of this study were the geometric parameters used to describe the profile of Γ_c. The contact surface was divided into two parts:

– a part of length l, over which the initial gap is zero, l being the first design parameter and b denoting the length of the complementary part: $b = 100 - l$;

– a part over which the shape is exponential, the coefficient λ being the second design parameter.

The profile g of the interface can therefore be written as:

$$\begin{cases} g = (e^{-\lambda x} - e^{-\lambda b})/100 & \text{for} \quad x < l \\ g = 0 & \text{for} \quad x \geq l \end{cases}$$

In order to reduce edge effects, we introduced two objective functions: the maximum normal stress at the contact interface f_1, and the standard deviation of the pressure distribution f_2.

$$f_1 = p_{max} \qquad \text{et} \qquad f_2 = \Big(\frac{1}{m}\sum_{i=1}^{m}(p_i - \overline{p})^2\Big)^{\frac{1}{2}}$$

where m is the number of nodes at the interface, p_i the pressure at node i, and $\overline{p}$ is the average pressure over Γ_c.

The parametric study was carried out using the multiresolution strategy. Figure 5.24 represents the cubic interpolation of the values of the two objective functions calculated for all the sets of parameters. Multiobjective optimization problems usually have a number of solutions called non-dominated solutions or optimum solutions in the Pareto sense [MIE 99]. Each optimum solution in the Pareto sense corresponds to the best possible compromise among the objectives. The Pareto front, defined as the set of the optima in the Pareto sense, is represented in Figure 5.24(a). Starting from a point belonging to the Pareto frontier, if one objective function is improved, the other is necessarily degraded. The formal definition of Pareto optimiality is given in Chapter 7. Figure 5.24 also shows the point corresponding to the case where the contact interface is a perfect plane ($l = 100\ mm$). As anticipated, this solution is quite distant from the Pareto set.

The first step, i.e. the generation of the metamodel (Figure 5.24) required 250 calculations. The first evaluation took 391 iterations and 366 s. The CPU time for the entire calculation was $11,778$ s, which corresponds to a significant gain, equal to $(250 \times 366)/11,778 = 7.8$. This gain is quite satisfactory because the variation in the design parameters only concerns interface parameters and, therefore, the operators of the substructures (stiffness matrix and homogenized operator) did not need to be re-evaluated for each calculation. The optimization of the metamodel was performed

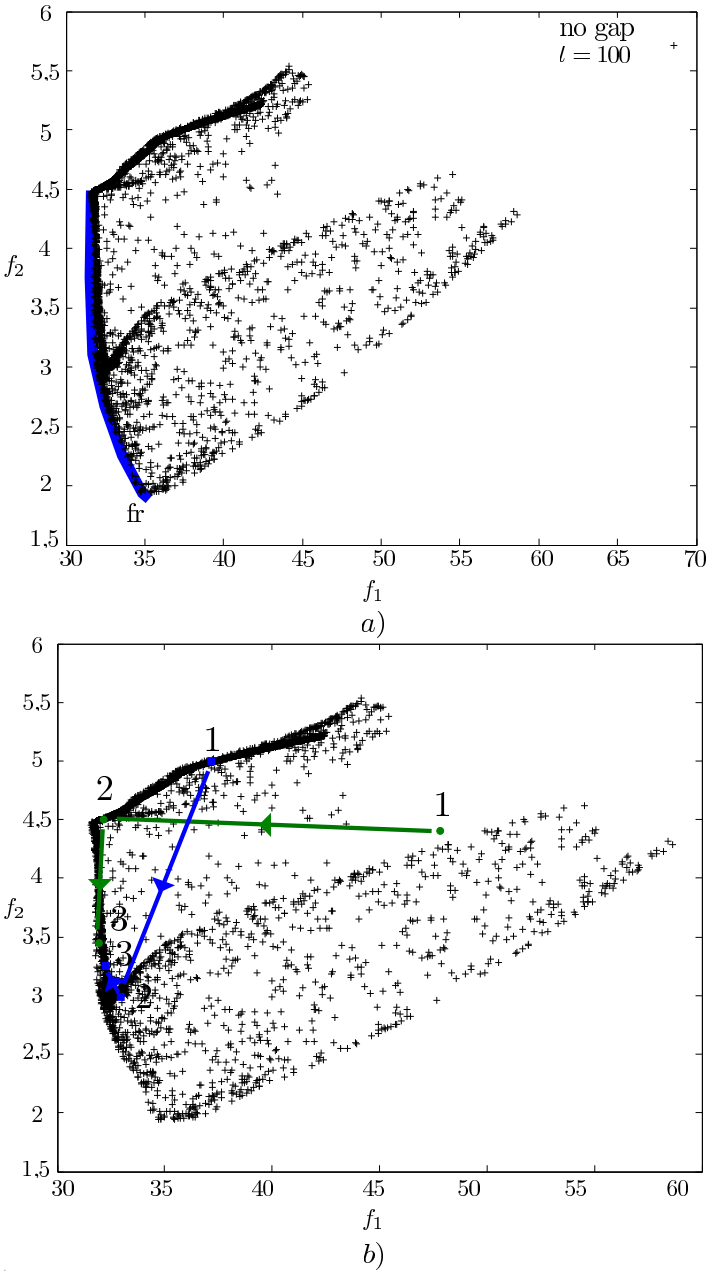


Figure 5.24. *Pareto diagram obtained from the metamodel.*

first in order to obtain the approximate minimum. The corresponding optimization problem is given by:

$$\begin{gathered} f_1(x^*) = min\{ f_1 = p_{max} \mid x \in \mathbf{S}\} \\ f_2(x^*) = min\{ f_2 = (1/m \sum_{i=1}^{m} (p_i - \overline{p})^2)^{\frac{1}{2}} \mid x \in \mathbf{S}\} \\ \mathbf{S} = \{0 \leq l \leq 90\ ,\ 0.01 \leq \lambda \leq 0.29\} \end{gathered} \qquad [5.28]$$

The optimization of the metamodel only took a few seconds and gave a good approximation of the solution (relatively close to the Pareto frontier). This approximate solution was then used to initialize the optimization of the mechanical model (Step 2 in Figure 5.11). Table 5.2 shows the design parameters for the two consecutive steps of the optimization and for two initial pairs, and Figure 5.24(b) gives the values of the corresponding objective functions. The resulting points at the end of the complete optimization process are optimum in the Pareto sense. The gain with multiresolution in terms of CPU time is also given in Table 5.2 for the two different initial points. These gains are very good because the optimization process required the calculation of numerous gradients, and the calculation of each gradient took only 2 s, whereas without multiresolution it took 686 s.

Initial parameters	approximate optima/ initial parameters for step 2	Exact optimum	Gain
Point 1	Point 2	Point 3	
$l^0 = 78\ mm$	$l = 4.5\ mm$	$l^* = 5.82\ mm$	
$\lambda^0 = 0.2$	$\lambda = 0.053$	$\lambda^* = 0.070$	44
$l^0 = 63\ mm$	$l = 49\ mm$	$l^* = 52\ mm$	
$\lambda^0 = 0.013$	$\lambda = 0.112$	$\lambda^* = 0.078$	41

Table 5.2. *The optimization data*

Figure 5.25 shows the profiles of the contact surfaces for the two optimum points ($l^* = 5.82\ mm$, $\lambda^* = 0.070$) and ($l^* = 52\ mm$, $\lambda^* = 0.078$) along with the distribution of the normal pressure over the interface. This distribution is also given in the case where the interface is exactly planar (with no initial gap). The optimum distributions (i.e. those which correspond to the optimum points in the Pareto sense) are very regular.

Finally, Table 5.3 gives the total gain obtained with the multiresolution strategy for the two sets of initial parameters.

Step	Number of simulations			CPU time
1	First calculation of the metamodel		1 250	366 s 11,778 s
2 First set of parameters	First calculation Optimization procedure	80	1 9 calls to the function 71 gradient evaluations	686 s 464 s 100 s
2 Second set of parameters	First calculation Optimization procedure	176	1 27 calls to the function 149 gradient evaluations	686 s 1,682 s 568 s
	Gain1 (First set of parameters) **Gain2** (Second set of parameters)			11.2 14.4

Table 5.3. *Gain obtained with the multiresolution strategy during the complete optimization scheme.*

The gains obtained for the two sets of parameters were calculated as follows:

$$Gain1 = \frac{250 \times 366 + 80 \times 686}{11,778 + 686 + 464 + 100} = 11.2$$

$$Gain2 = \frac{250 \times 366 + 176 \times 685}{11,778 + 685 + 1,682 + 568} = 14.4$$

5.9.3. *Example 3: optimization of the tightening of a bolt*

Let us consider the cast iron manifold in Figure 5.26 ($E = 120,000$ MPa, $\nu = 0.3$, $\alpha = 10\ 10^{-6}\,K^{-1}$) supported by a fixed aluminum base ($E = 70,000$ MPa, $\nu = 0.3$, $\alpha = 20\ 10^{-6}\,K^{-1}$). A gasket, which is also made of aluminum, makes the assembly of these two parts leakproof. These are joined together by seven steel bolts and nuts ($E = 200,000$ MPa, $\nu = 0.3$, $\alpha = 20\ 10^{-6}\,K^{-1}$). The meshes defined at the contact surfaces are fully compatible. The 17 substructures being considered are: the manifold, the base, the gasket, the seven bolts, and the seven nuts. The interfaces are represented in Figure 5.27. The interfaces of interest, over which the parameters vary, are the interfaces between the nuts and the bolts. We assume that the same bolts of the three independent groups are identically tightened. Thus, we have a problem with only three independent variables denoted by pre_1, pre_2, and pre_3.

The external solicitations were applied in three steps. First, the seven bolts were tightened (prestressing step). Then a mechanical loading was applied incrementally

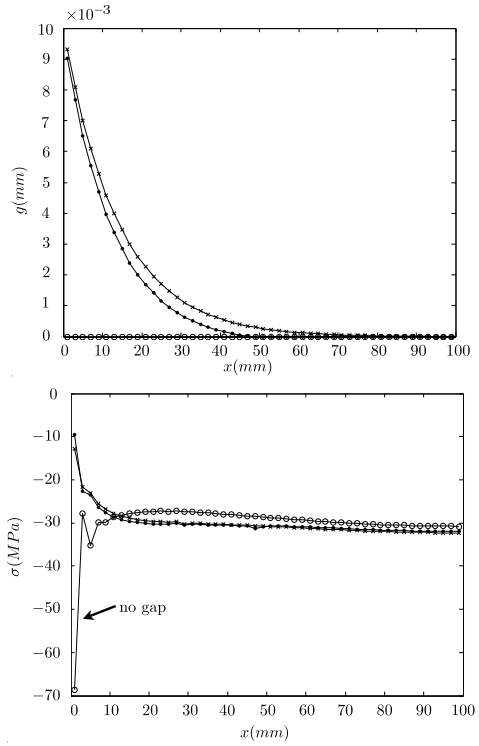


Figure 5.25. *Profile of the contact interface (top) and distribution of the contact pressure over interface Γ_c (bottom).*

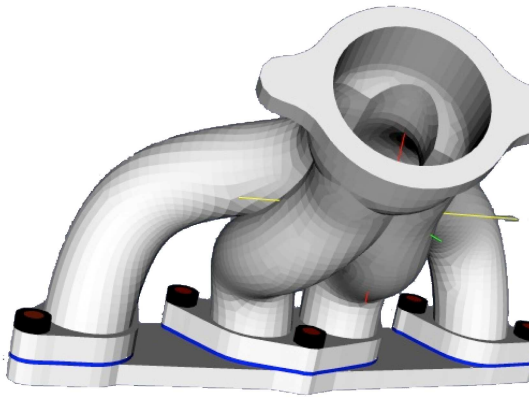


Figure 5.26. *The manifold model.*

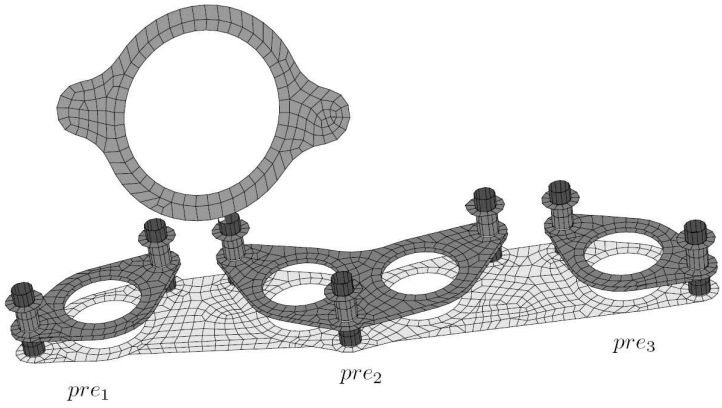


Figure 5.27. *Interfaces.*

over the "head" of the manifold in the z-direction (see Figure 5.26). Finally, while maintaining the other two loading cases, the whole assembly was heated progressively up to a 600 C temperature difference. The evolution of the loading is represented schematically in Figure 5.28. The tightening of the bolts was applied by setting a negative axial gap between the bolt and the nut (see Figure 5.29). These gaps were considered to be the design variables of the problem.

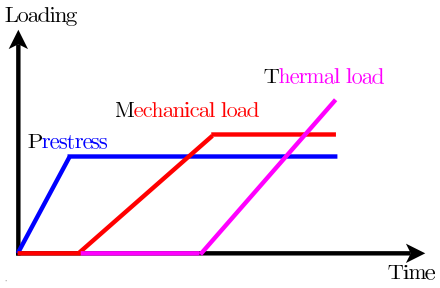


Figure 5.28. *Evolution of the loading.*

Figure 5.30 shows the Von Mises' stress in the assembly at the end of each loading step for prestress values $pre_1 = 0.04$ mm, $pre_2 = 0.02$ mm, and $pre_3 = 0.055$ mm. At the end of the bolt-tightening step (Figure 5.30 a), the gasket is crushed against the base and the stresses are localized near the bolts. At the end of the mechanical loading (Figure 5.30(b), the manifold becomes very slightly detached, resulting in loss of contact between the gasket and the two parts. Heating (Figure 5.30(c) causes global expansion of the manifold and significant swelling of the gasket.

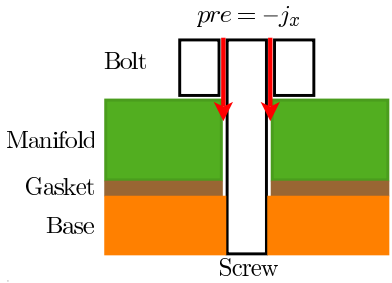


Figure 5.29. *Prestressing of the assembly bolt.*

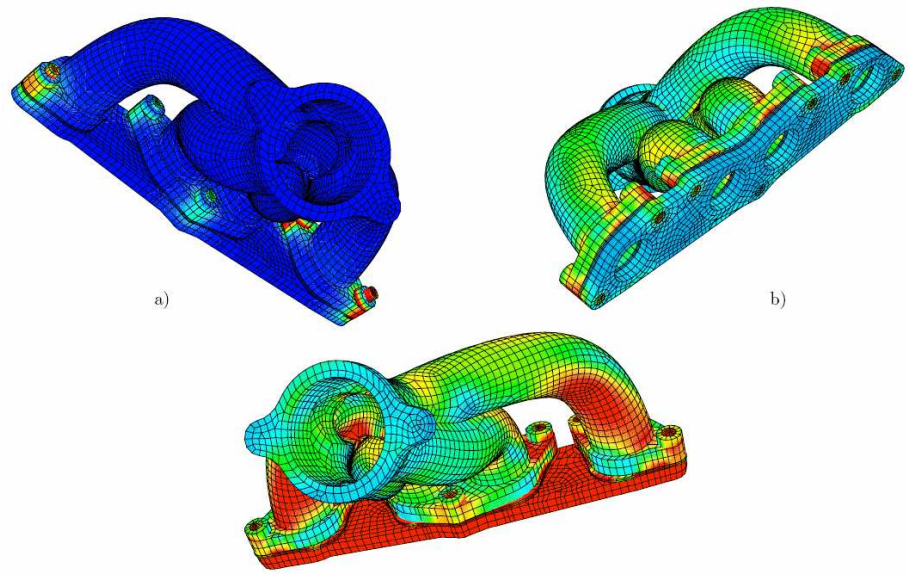


Figure 5.30. *Von Mises' stresses.*

The first step of the complete optimization process (Figure 5.24) required a parametric study in which the three prestresses were varied simultaneously. In Figure 5.31, preload pre_3 is fixed and the average contact pressure between the gasket and the manifold is shown as a function of the evolution of the other two parameters at the end of the two mechanical and thermal loading steps.

Generation of the metamodel required 343 ($7 \times 7 \times 7$) calculations. At this stage in the process, the gain was already 7. This result is very satisfactory and is of the same

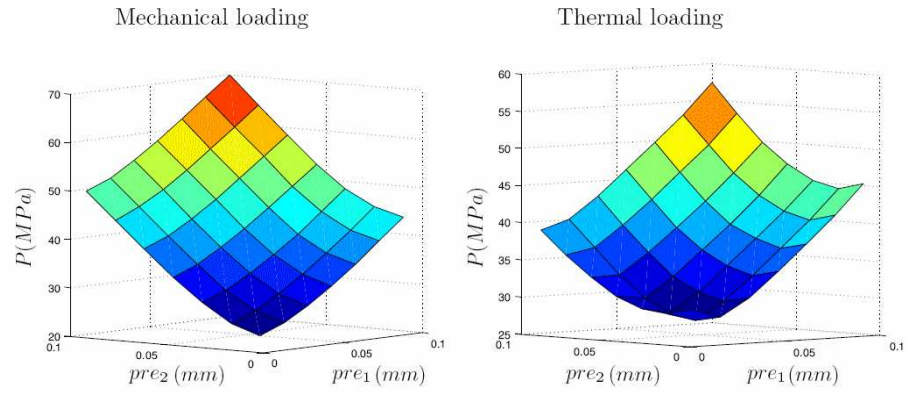


Figure 5.31. *Evolution of the average contact pressure as a function of the preload in the bolts* $pre_3 = 0.025$ *mm*

order of magnitude as that of the previous 2D example (gain of 7.8 for the parametric study).

This reached the limits of the multiresolution strategy in this context: indeed, the previous calculations were all performed with a regular grid in the parameter space and, from a numerical point of view, the implementation was done by imbricating three loops over these values of the parameters. Thus, a calculation was reinitialized from the results of the previous calculation. Clearly, restarting from the previous calculation is not the best strategy. We can easily imagine that among the set of the points already calculated there is a particular reinitialization point which is better than the others (see 5.9.4).

At the end of the metamodel generation phase, the optimization was then carried out using the metamodel. From that point, so as not to increase the computing costs (particularly in the optimization phase of the complete model), the parameter pre_3 was set to 0.025 mm, and we sought the optimum tightening value which minimizes the average contact pressure between the gasket and the manifold. Thus, the optimization problem can be expressed as follows:

$$f(x^*) = min\{f = p_{avg} \mid x \in \mathbf{S}\}$$
$$\mathbf{S} = \{0.01 \leq pre_1 \leq 0.1\ ,\ 0.01 \leq pre_2 \leq 0.1\}$$

The starting point of the optimization of the metamodel was:

$$(pre_1^{(0)}, pre_2^{(0)}) = (0.09, 0.08).$$

The calculation required 37 calls to the cost function and took only a few seconds to converge towards an approximate minimum which was quite acceptable $(pre_1, pre_2) = (0.022, 0.025)$. Then, the optimization was performed over the complete model using the converged solution from the optimization of the metamodel in the initialization. The time taken for calculation of the gradients is estimated to have been 9.33 s, while the time for the first calculation was 1543 s. Finally, the optimum solution was

$$(pre_1^*, pre_2^*) = (0.01, 0.01)$$

and the global gain obtained with the multiresolution strategy was about 22.

5.9.4. *Example 4: Multiresolution/metamodel coupling*

The multiresolution aspect is fundamental in reducing computing costs. The multiresolution strategy is used in:

– generating the metamodel;

– optimizing on the complete model, particularly in the calculation of the numerical gradients.

Let us now present a strategy for improving the multiresolution by seeking the best point for reinitializing the LATIN algorithm. Indeed, at a given time in the metamodel generation phase or in the optimization phase of the complete model, we can have access to several already calculated solutions. Thus, in performing a new calculation, we can reasonably assume that among the previously calculated solutions one is likely to maximize the cost reduction for this new calculation. Several ideas are to be considered, one of which couples this intelligent search with the use of an enriched metamodel, which is constructed gradually as the optimization proceeds. We will compare the following strategies using an academic example and an industrial case:

– *Initial point*: the i^{th} calculation is reinitialized from the results of the first calculation.

– *Previous point*: the i^{th} calculation is reinitialized from the results of calculation $i - 1$.

– *Parameter distance*: the i^{th} calculation is reinitialized from the results of the closest calculation, in the sense of a distance in the parametric space.

– *Response distance*: the i^{th} calculation is reinitialized from the results of the closest calculation (with respect to a response), in the sense of a distance in the space of this response.

The time gain obtained compared to using a classical resolution method is defined as the ratio of the time without multiresolution to the time with multiresolution:

$$Gain = \frac{number\ of\ calculations\ \times\ CPU\ time\ of\ the\ first\ calculation}{CPU\ time\ with\ the\ multiresolution\ strategy} \qquad [5.29]$$

The time corresponding to the first calculation is taken as the reference, and we assume that the time required for each parameter variation is the same as the time for the first calculation.

5.9.4.1. *Parametric study of contact with friction in an assembly*

We illustrate our approach and analyze the gain in terms of computing time with two optimization cases related to connection parameters in assemblies. The first case, an academic example with two parameters, concerns the contact properties of different parts of an assembly. To complement this didactic example, we also present a second, industrial example with six parameters relative to a shaft/pinion assembly using a shrink disc.

5.9.4.2. *The academic example*

Let us consider three identical square structures in contact, with friction (see Figure 5.32). The parametric study concerns the friction coefficients μ_1 and μ_2 of the two contact surfaces. First, Ω_3 is preloaded with a vertical pressure F_1; then, the middle structure Ω_2 is pushed against a stop with a horizontal force F_2. We carried out the incremental calculation for a drawing of $N = k^2$ different pairs of parameters with $k \in \{1, \ldots, 10\}$ – organized as a complete factorial DOE – and obtained the horizontal reaction F of the stop under the full prescribed lateral loading. We then performed a second random drawing of N points with $N \in \{9, 16, 25, 36, 49, 64, 81, 100\}$. Tables 5.5 and 5.4 show that the coupling of the multiresolution method with an intelligent search strategy for the reinitialization point led to a gain in computing time of between 1.5 and about 6. With a refined mesh of the domain being studied and a small number of parameters (two in this case), the distance in the space of the parameters seems to be the most relevant criterion.

In this example, it is understood that to reinitialize the LATIN algorithm from the results of the closest calculation with respect to a response, in the sense of a distance in the space of this response, leads to choosing a calculation on an isovalue which can be far from the point of calculation considered in the sense of a distance in the parametric space.

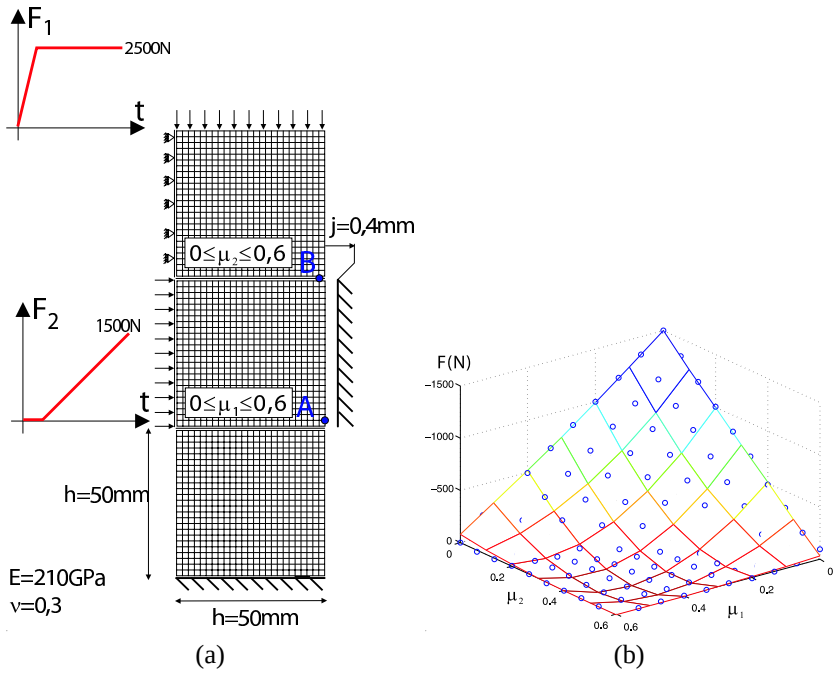


Figure 5.32. *(a) FE model, and (b) resulting metamodel of F as a function of μ_1 and μ_2.*

Strategy	number of iteration	CPU time (s) with multiresolution	CPU time (s) without multiresolution	Gain
initial point	11621	1778	2667	1.50
previous point	3316	529	2697	5.09
parameter distance	2691	431	2673	6.19

Table 5.4. *Gain for a DOE of 100 simulations*

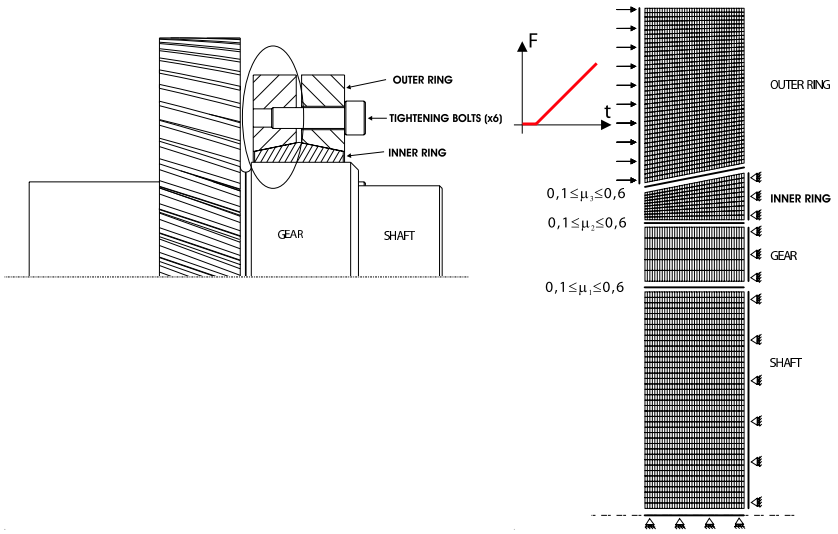
5.9.4.3. *Analysis of a shaft and pinion connected by a shrink disc*

The second example concerns the complete binding of a shaft/pinion assembly using a frictional connection with a shrink disc: see Figure 5.33(a).

The transmissible torque and the transmissible axial load depend on the coefficient of friction between the shaft and the hub, the fit clearance, and the shaft diameter. The evolution of the torque and axial load which can be transmitted by the connection was studied with respect to six parameters (the loads and friction parameters among

Strategy	number of iteration	CPU time (s) with multiresolution	CPU time (s) without multiresolution	Gain
initial point	1841	297	1247	4.20
previous point	1371	190	891	4.69
parameter distance	1311	140	846	6.02
response distance	1611	174	878	5.03

Table 5.5. *Gain for a Latin hypercube sampling of 25 simulations*



(a) (b)

Figure 5.33. *(a) Assembly, and (b) FE model.*

the different parts in contact) using an axisymmetric model (see Figure 5.33b). Table 5.6 indicates a greater time gain, equal to about 11, for a Latin hypercube sampling (LHS) of 100 simulations. The criterion based on the metamodel led to a gain which is comparable to that of the distance in the parametric space.

Comparison of a DOE with an LHS sampling shows that an orthogonal array decreases the computing time. Table 5.7 indicates a greater time gain, equal to about 17, for a three-level orthogonal array of 81 simulations.

The multiresolution, which had already shown its ability to reduce computing costs in relation to optimization strategies, can be significantly further improved using an

Strategy	number of iteration	CPU time (s) with multiresolution	CPU time (s) without multiresolution	Gain
initial point	2656	540	3912	7.24
previous point	2811	572	3890	6.81
parameter distance	1731	357	3930	11.01
response distance	1746	356	3898	10.94

Table 5.6. *Gain for a Latin hypercube sampling of 100 simulations.*

Strategy	number of iteration	CPU time (s) with multiresolution	CPU time (s) without multiresolution	Gain
initial point	5166	1031	3593	3.48
previous point	2631	539	3620	6.71
parameter distance	1056	216	3577	16.53
response distance	996	206	3549	17.26

Table 5.7. *Gain for a three-level DOE sampling with 81 simulations*

intelligent search for the optimum reinitialization point. Comparison of different criteria shows that the gain can be increased and that it is relevant to couple this intelligent search with the metamodel constructed during the optimization procedure. A mixed criterion involving the distance in the parametric space and the distance in the space of the responses should enable this gain to be increased even further. The choice of the size and type of the sampling also constitutes a way of improving the method. The evolution of the gain for larger problems must be considered from the point of view of both the number of parameters and the number of simulations.

5.9.5. *Example 5: Supersonic business jet optimization*

5.9.5.1. *Test-case description*

We consider the drag minimization of a supersonic business jet at a Mach number of $M_\infty = 1.7$ and angle of attack $\alpha = 1^o$ subject to a constraint on the lift, volume, and thickness. The constraints are implemented by adding penalty terms to the cost function. The governing equations are the Euler equations of inviscid compressible flow; hence the drag is only composed of lift-induced drag and wave drag. The wave drag has contributions from lift and volume; a reduction in drag can be obtained just by reducing the volume. Since in practice the volume of the wing has to be maintained for structural and other reasons, we impose a constraint on the volume in the cost function through a volume penalty term. The wings of supersonic aircraft are very thin in order to reduce wave drag; the optimization must not reduce the thickness of

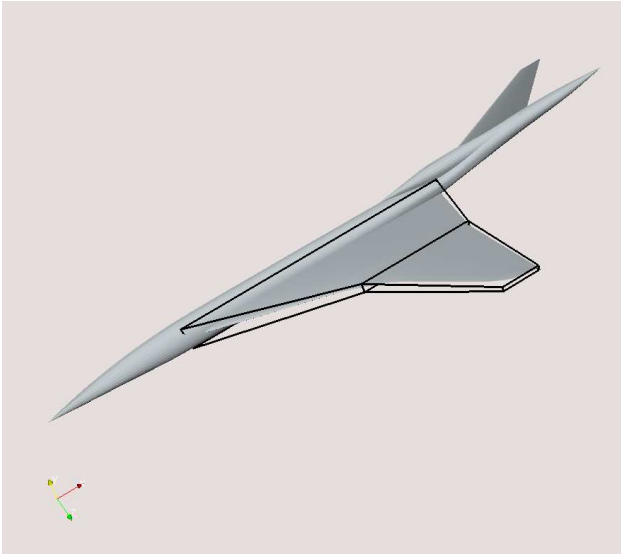


Figure 5.34. *FFD box for supersonic business jet.*

the wing since this affects its structural strength. Hence a penalty term which controls the thickness is added to the cost function. The cost function used is given below.

$$J = \frac{C_d}{C_{d_o}} + 10^4 \max\left(0, 0.999 - \frac{C_l}{C_{l_o}}\right) + 10^3 \max(0, V_o - V) + I_p \qquad [5.30]$$

where C_d = drag coefficient, C_l = lift coefficient, V = volume of the wing, and I_p = a penalty term to control the thickness. The quantities with subscript "o" indicate values corresponding to the reference or starting shape. The penalty term I_p is computed as follows. A box is inserted inside the reference wing. When the wing grid is deformed, some points of the grid lying on the wing may go inside this box. The term I_p is computed as

$$I_p = 1000 \frac{\text{number of grid points on wing surface lying inside the box}}{\text{total number of grid points on the wing surface}} \qquad [5.31]$$

This term approximately models the fraction of the wing surface that penetrates the inner box and thus penalizes the cost function if the wing thickness becomes too small. The CFD computations are performed on an unstructured grid with 37,375 nodes and 184,249 tetrahedra using a finite volume solver developed at INRIA and described in [DER 92].

5.9.5.2. *FFD parametrization*

A critical issue in parametric shape optimization is the choice of the shape parametrization. The objective of the parametrization is to describe the shape, or the shape modification, by a set of parameters which are considered as design variables during the optimization procedure. The free-form deformation (FFD) technique [SED 86] is adopted in the present study, since it provides an easy and powerful framework for the deformation of complex shapes, such as those encountered in aerodynamics. It allows the deformation of an object in a 2D or 3D space, regardless of the representation of this object. Instead of manipulating the surface of the object directly, using classical B-splines or Bézier parametrization of the surface, FFD techniques define a deformation field over the space embedded in a lattice which is built around the object. By transforming the space coordinates inside the lattice, the FFD technique deforms the object, regardless of its geometric description.

More precisely, consider a 3D hexahedral lattice embedding the object to be deformed. Figure (5.34) shows an example of such a lattice built around a realistic wing. A local coordinate system (ξ, η, ζ) is defined in the lattice, with $(\xi, \eta, \zeta) \in [0,1] \times [0,1] \times [0,1]$. During the deformation, the displacement Δq of each point q inside the lattice is defined by a third-order Bézier tensor product:

$$\Delta q = \sum_{i=0}^{n_i} \sum_{j=0}^{n_j} \sum_{k=0}^{n_k} B_i^{n_i}(\xi_q) B_j^{n_j}(\eta_q) B_k^{n_k}(\zeta_q) \Delta P_{ijk}. \qquad [5.32]$$

$B_i^{n_i}$, $B_j^{n_j}$, and $B_k^{n_k}$ are the Bernstein polynomials of order n_i, n_j, and n_k (see for instance [FAR 89]):

$$B_p^n(t) = C_n^p \, t^p \, (1-t)^{n-p}. \qquad [5.33]$$

$(\Delta P_{ijk})_{0 \le i \le n_i, 0 \le j \le n_j, 0 \le k \le n_k}$ are weighting coefficients, or control point displacements, which are used to monitor the deformation and are considered as design variables during the shape-optimization procedure.

The FFD parametrization is only built around the wing, as shown in Figure (5.34), with ξ, η, and ζ in the chordwise, spanwise, and thickness directions respectively. The lattice is chosen to fit the planform of the wing as closely as possible. The leading and trailing edges are kept fixed during the optimization by freezing the control points that correspond to $i = 0$ and $i = n_i$. The control points corresponding to $k = n_k$, which control the displacement of the wing tip, are kept constant. Moreover, control points are only moved vertically. The parametrization corresponds to $n_i = 6$, $n_j = 1$, and $n_k = 2$ and leads to $(7-2) \times 2 \times 2 = 20$ degrees of freedom.

Method	100% CFD	30% CFD	20% CFD	10% CFD	Adaptive
Cost	0.9212	0.9144	0.9148	0.9097	0.9183
Iterations	216	400	500	500	500
CFD eval.	25920	15240	12960	7080	6002

Table 5.8. *Results of PSO for supersonic business jet.*

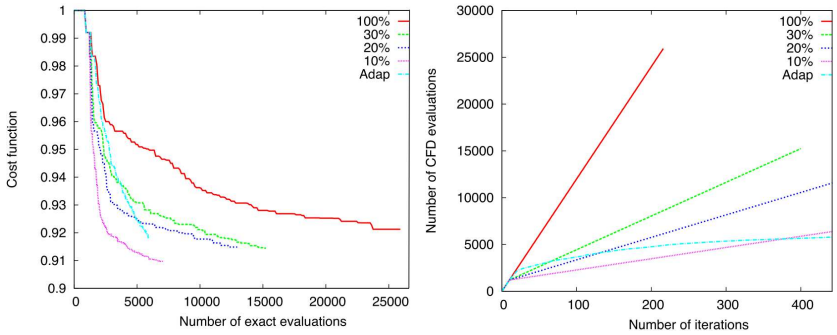


Figure 5.35. *Optimization of supersonic business jet: (a) evolution of cost function, and (b) number of CFD evaluations*

5.9.5.3. *Metamodel-assisted PSO optimization*

We perform the shape optimization using CFD as the exact model. The metamodel is used with 10%, 20%, and 30% CFD evaluations and the adaptive prescreening criteria. The local database is constructed with 40 nearest points from the database. When metamodels are used, more iterations are performed in PSO since the total number of exact evaluations is small. The results are given in Table (5.8) and Figure (5.35). Using a metamodel and IPE gives the same level of cost function as that obtained with full CFD evaluations. Both prescreening criteria give similar levels of cost functions but the 10% evaluations and adaptive criterion are the more efficient. Figure (5.35-a) shows the evolution of the cost function as a function of the number of CFD evaluations, while Figure (5.35(b) shows the number of CFD evaluations as a function of the PSO iterations. Finally, Figure (5.36) shows the shapes for the initial configuration, the CFD-optimized configuration, and the CFD + metamodel optimized configuration. It can be seen that the optimized shapes obtained using CFD alone and with metamodels are similar, indicating that the use of metamodels leads to similar results.

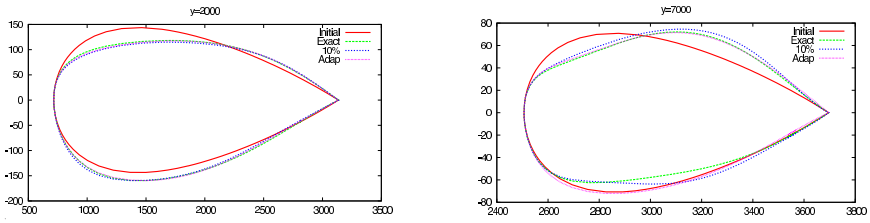


Figure 5.36. *Wing shapes for supersonic business jet at different spanwise stations.*

5.9.6. *Example 6: Transonic wing optimization*

5.9.6.1. *Test-case description*

The test-case considered here corresponds to the optimization of the shape of the wing of a business aircraft (courtesy of Piaggio Aero Ind.) for a transonic regime. The test-case is described in depth in [AND 03]. The overall wing shape can be seen in Figure (5.37). The free-stream Mach number is $M_\infty = 0.83$ and the incidence $\alpha = 2°$. Initially, the wing section corresponds to the NACA 0012 airfoil.

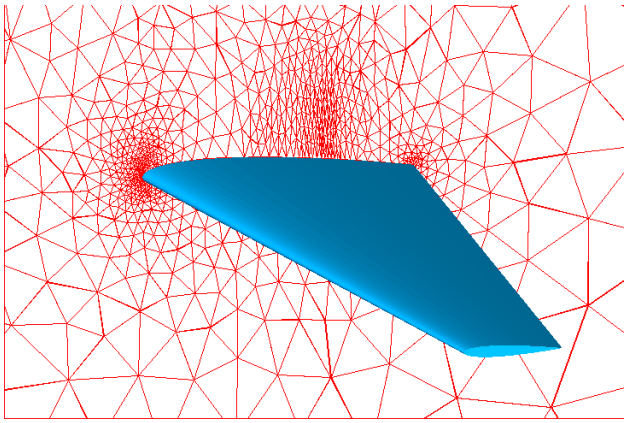


Figure 5.37. *Transonic wing: initial wing shape (blue) and mesh in the symmetry plane (red).*

The goal of the optimization is to reduce the drag coefficient C_d subject to the constraint that the lift coefficient C_l should not decrease more than 0.1%. The constraint is taken into account using a penalization approach. The resulting cost function is:

$$J = \frac{C_d}{C_{do}} + 10^4 \max(0, 0.999 - \frac{C_l}{C_{lo}}) + 10^3 \max(0, V_o - V) \qquad [5.34]$$

C_{do} and C_{lo} are respectively the drag and lift coefficients corresponding to the initial shape (NACA 0012 section) and V_o is the wing volume. For the CFD computations, an unstructured mesh, composed of 31,124 nodes and 173,445 tetrahedral elements, is generated around the wing, including a refined area in the vicinity of the shock (Figure (5.37)).

5.9.6.2. *FFD parametrization*

The FFD lattice is built around the wing with ξ, η, and ζ in the chordwise, spanwise, and thickness directions, respectively. The lattice is chosen to fit the planform of the wing. The leading and trailing edges are kept fixed during the optimization by freezing the control points that correspond to $i = 0$ and $i = n_i$. Moreover, the control points are only moved vertically. The parametrization corresponds to $n_i = 6$, $n_j = 1$, and $n_k = 1$ and gives $(7 - 2) \times 2 \times 2 = 20$ degrees of freedom.

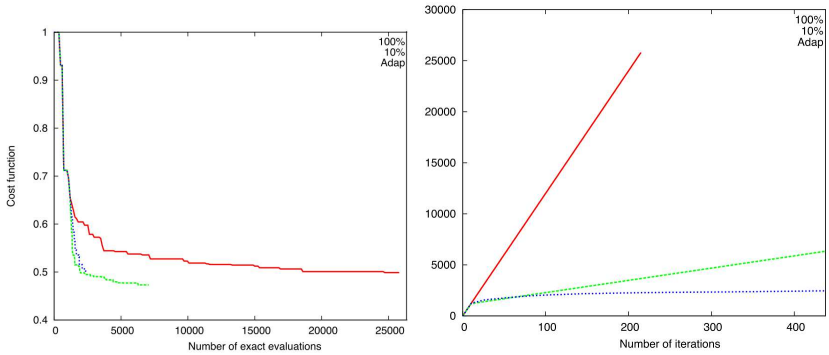
5.9.6.3. *Metamodel-assisted PSO optimization*

The optimization is performed using PSO with 120 particles and the same set of parameters as in the previous section. The local metamodels are constructed using 40 nearest neighbors from the database. In the case of metamodel-assisted PSO, 500 iterations are performed. Table (5.9) shows the results of optimization. The metamodel-assisted PSO is found to yield a cost function similar to the full CFD case, while the number of CFD evaluations is significantly small. Figure (5.38(a)) shows the evolution of the cost function with the number of CFD evaluations. Both the prescreening criteria are able to yield reductions in cost function comparable to those obtained in the exact evaluations case. The 10% exact evaluations case finds a lower cost function than the adaptive case, or even the 100% exact case, because we are able to perform more PSO iterations.

Figure (5.38(c)) shows the variation in the number of CFD evaluations with the iteration number. As in the case of a supersonic business jet, the CFD evaluation count for the adaptive case grows very slowly and asymptotes to a nearly constant value, indicating that the number of CFD evaluations goes to zero as the PSO iterations increase. Finally, Figure (5.39) shows a comparison of the airfoil shapes at different spanwise locations. The shapes obtained with metamodel-assisted PSO are quite close to those obtained with 100% CFD evaluations. In particular, the shape of the upper surface is more critical since the shock is found on this side of the airfoil. We notice that metamodel-assisted optimization leads to very similar shapes on the upper surface.

Method	100% CFD	10% CFD	Adaptive
Cost	0.4987	0.4730	0.5018
Iterations	215	500	500
CFD eval.	25800	7080	2511

Table 5.9. *Optimization of a transonic wing.*



(a) (b)

Figure 5.38. *Optimization of transonic wing: (a) evolution of cost function, and (b) number of CFD evaluations.*

5.10. Conclusion

We have presented the main multilevel optimization strategies. The proposed classification enabled us to group the main methods found in the literature into the following categories:

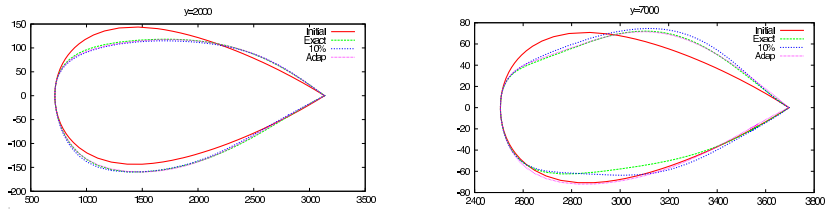


Figure 5.39. *Wing shapes for transonic wing at different spanwise stations.*

– parallel model optimization;
– optimization with multiple parameter levels
  - sequential optimization;
  - iterative optimization;
– optimization with multiple model levels
  - hierarchical optimization;
  - imbricated optimization;

All these strategies, thanks to the modification of the initial problem, enable the cost of a global optimization to be reduced, but sometimes at the expense of losing the convergence towards the optimum solution. It is therefore fundamental to modify the initial model in a conservative way in order to guarantee convergence.

The use of global optimization methods (such as evolutionary strategies or particle swarm optimization) on the basis of high-fidelity solvers is still an issue, despite the growth of computational facilities. The development of multilevel methods is certainly a key element in solving an optimum design problem using refined models. We can imagine all sorts of methods based on databases, on coarse meshes, or on simplified physical approaches. The literature shows clearly that these different approaches work properly in numerous test-cases.

However, the real challenge today is to develop algorithms using these multilevel models in an *adaptative* and *automatic* way, without the need for the user to set the parameters manually. Such an algorithm will have to determine the modeling level required for the problem considered automatically, and adapt this level throughout the optimization process.

5.11. Bibliography

[ALE 00] Alexandrov N. M., Lewis R. B., Analytical and computational aspects of collaborative optimization, Report , NASA/TM-2000-210104, 2000.

[ALL 05a] Allaire G., Jouve F., "A level-set method for vibration and multiple loads structural optimization", *Computer Methods in Applied Mechanics and Engineering*, vol. 194, p. 3269–3290, 2005.

[ALL 05b] Allix O., Feissel P., Nguyen H. M., "Identification strategy in the presence of corrupted measurements.", *Engineering Computations*, vol. 22, num. 5/6, p. 487–504, 2005.

[AND 03] Andreoli M., Janka A., Desideri J.-A., Free-form deformation parameterization for multilevel 3D shape optimization in aerodynamics, Report num. 5019, INRIA, November 2003.

[BEN 95] Bendsoe M.-P., *Optimization of Structural Topology, Shape and Material*, Springer-Verlag, Heidelberg, 1995.

[BOU 03] Boucard P.-A., Champaney L., "A suitable computational strategy for the parametric analysis of problems with multiple contact.", *International Journal for Numerical Methods in Engineering*, vol. 57, p. 1259–1282, 2003.

[BOU 04] Boucard P.-A., Champaney L., "Approche multirésolution pour l'étude paramétrique d'assemblages par contact et frottement.", *Revue Européenne des Eléments Finis*, vol. 13, num. 5/7, p. 437–448, 2004.

[BOU 07] Boucard P.-A., Buytet S., Guidault P.-A., "Une stratégie multi-échelle pour l'étude paramétrique de détails géométriques au sein de structures en contacts multiples", *Revue Européenne de Mécanique Numérique*, vol. 16, num. 8, p. 1011–1036, 2007.

[BUC 05] Buche D., Schraudolph N. N., Koumoutsakos P., "Accelerating evolutionary algorithms with Gaussian process fitness function models", *IEEE Tran. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 35, num. 2, 2005.

[CHA 95] Chattopadhyay A., McCarthy T.-R., Pagaldipti N., "Multilevel decomposition procedure for efficient design optimization of helicopter rotor blades", *AIAA Journal*, vol. 35, p. 223–230, 1995.

[CHA 97] Champaney L., Cognard J., Dureisseix D., Ladevèze P., "Large scale applications on parallel computers of a mixed domain decomposition method.", *Computational Mechanics*, vol. 19, p. 253–263, 1997.

[CHA 99] Champaney L., Cognard J., Ladevèze P., "Modular analysis of assemblages of three-dimensional structures with unilateral contact conditions", *Computers and Structures*, vol. 73, p. 4249–266, 1999.

[CHE 05] Chen T. Y., Yang C. M., "Multidisciplinary design optimization of mechanisms", *Advances in engineering software*, vol. 36, p. 301–311, 2005.

[CON 02] Conceiça Antonio C., "A multilevel genetic algorithm for optimization of geometrically nonlinear stiffened composite structures", *Structural Multidisciplinary Optimization*, vol. 24, p. 372–386, 2002.

[DER 92] Dervieux A., Desideri J. A., Compressible flow solvers using unstructured grids, Research Report num. 1732, INRIA, June 1992.

[DES 07] Desmorat B., "Structural rigidity optimization with frictionless unilateral contact", *International Journal of Solids and Structures*, vol. 44, num. 3–4, p. 1132–1144, 2007.

[DUV 06] Duvigneau R., Chaigne B., Desideri J.-A., Multi-level parameterization for shape optimization in aerodynamics and electromagnetics using particle swarm optimization, Research Report num. RR-6003, INRIA, Sophia Antipolis, 2006.

[ELS 91] El-Sayed M. E., Hsiung C.-K., "Optimum structural design with parallel finite element analysis", *Compututers and Structures*, vol. 40, num. 6, p. 1469–1474, 1991.

[EMM 06] Emmerich M., Giannakoglou K., Naujoks B., "Single- and multi-objective evolutionary optimization assisted by Gaussian random field metamodels", *IEEE Trans. Evol. Comput.*, vol. 10, num. 4, p. 421–439, 2006.

[ENG 04] Engels H., Becker W., Morris A., "Implementation of a multi-level optimisation methodology within the e-design of a blended wing body", *Aerospace Science and Technology*, vol. 8, p. 145–153, 2004.

[FAR 89] FARIN G., *Curves and surfaces for computer-aided geometric design*, Academic Press, 1989.

[FOU 02] FOURIE P., GROENWOLD A., "The particle swarm optimization in size and shape optimization", *Structural and Multidisciplinary Optimization*, vol. 23, num. 4, 2002.

[GIA 02] GIANNAKOGLOU K. C., "Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence", *Prog. Aero. Sci.*, vol. 38, p. 43–76, 2002.

[GUI 06] GUIDAULT P.-A., ALLIX O., CHAMPANEY L., CORNUAULT C., "Une approche micro-macro pour le suivi de fissure avec enrichissement local", *Revue Européenne de Mécanique Numérique*, vol. 15, p. 187–198, 2006.

[JIN 05] JIN Y., "A comprehensive survey of fitness approximation in evolutionary computation", *Soft Computing*, vol. 9, num. 1, 2005.

[KEA 00] KEANE A.-J., PETRUZZELI N., "Aircraft wing design using GA-Based multi-level strategies", *Proceedings 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, CA, USA, p. A00-40–171, September 2000.

[KEN 95] KENNEDY J., EBERHART R., "Particle swarm optimization", *IEEE International Conference on Neural Networks*, Perth, Australia, 1995.

[KIM 00] KIM Y. Y., YOON G. H., "Multi-resolution multi-scale topology optimization : a new paradigm", *International Journal of Solids and Structures*, vol. 37, p. 5529–5559, 2000.

[KRA 03] KRAVANJA S., SORSAK A., KRAVANJA Z., "Efficient multilevel MINLP strategies for solving large combinatorial problems in engineering", *Optimization and engineering*, vol. 4, num. 1/2, p. 97–151, 2003.

[KRA 05] KRAVANJA S., SILIH S., KRAVANJA Z., "The multilevel MINLP optimization approach to structural synthesis : the simultaneous topology, material, standard and rounded dimension optimization", *Advances in engineering software*, vol. 36, p. 568–583, 2005.

[LAD 99] LADEVÈZE P., *Nonlinear Computational Structural Mechanics - New Approaches and non-Incremental Methods of Calculation*, Springer-Verlag, Berlin, 1999.

[LAD 01] LADEVÈZE P., LOISEAU O., DUREISSEIX D., "A micro-macro and parallel computational strategy for highly heterogeneous structures.", *International Journal for Numerical Methods in Engineering*, vol. 52, p. 121–138, 2001.

[LAD 02] LADEVÈZE P., NOUY A., LOISEAU O., "A multiscale computational approach for contact problems", *Computer Methods in Applied Mechanics and Engineering*, vol. 191, p. 4680–4891, 2002.

[LAD 03] LADEVÈZE P., NOUY A., "On a multiscale computational strategy with time and space homogenization for structural mechanics", *Computer Methods in Applied Mechanics and Engineering*, vol. 192, p. 3061–3088, 2003.

[LER 98] LE RICHE R., GAUDIN J., "Design of Dimensionally Stable Composites by Evolutionary Optimization", *Composite Structures*, vol. 41, p. 97–111, 1998.

[LI 05] LI W., LI Q., STEVEN G.-P., XIE Y. M., "An evolutionary shape optimization for elastic contact problems subject to multiple load cases", *Computer Methods in Applied Mechanics and Engineering*, vol. 194, p. 3394–3415, 2005.

[LIU 04] Liu B., Haftka R.-T., Watson L.-T., "Global-local structural optimization using response surfaces of local optimization margins", *Structural and Multidisciplinary Optimization*, vol. 27, num. 5, p. 352–359, 2004.

[MIC 92] Michalewics Z., *Genetic algorithms + data structures = evolutionary programs*, AI Series, Springer-Verlag, New York, 1992.

[MIE 99] Miettinen K. M., *Nonlinear Multiobjective Optimization*, Kluwer, Boston, 1999.

[MIL 07] Miller P., "Swarm behaviour", *National Geographic*, July 2007, available online at `http://www7.nationalgeographic.com/ngm/0707/feature5/`.

[MOË 99] Moës N., Dolbow J., Belytschko T., "A finite element method for crack growth without remeshing", *International Journal for Numerical Methods in Engineering*, vol. 46, p. 131–150, 1999.

[ONG 04] Ong Y. S., Nair P. B., Keane A. J., Wong K. W., "Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems", Jin Y., Ed., *Knowledge Incorporation in Evolutionary Computation*, Studies in Fuzziness and Soft Computing, Springer Verlag, 2004.

[OSH 88] Osher S., Sethian J.-A., "Fronts propaging with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations", *Journal of Computational Physics*, vol. 79, p. 12–49, 1988.

[ROB 99] Robinson G.-M., Keane A.-J., "A case for multi-level optimisation in aeronautical design", *Aeronautical Journal*, vol. 103, p. 481–485, 1999.

[ROU 98] Roux W.-J., Stander N., R.-T. H., "Response surface approximations for structural optimization.", *International Journal for Numerical Methods in Engineering*, vol. 42, p. 517–534, 1998.

[SAK 03] Sakata S., Ashida F., Zako M., "Structural optimization using Kriging approximation.", *Computer Methods in Applied Mechanics and Engineering*, vol. 192, p. 923–939, 2003.

[SED 86] Sederberg T., Parry S., "Free-form deformation of solid geometric models", *Computer Graphics*, vol. 20, num. 4, p. 151–160, 1986.

[SHI 98] Shi Y., Eberhart R., "A modified particle swarm optimizer", *International Conference on Evolutionary Computation*, 1998.

[STO 00] Stolarska M., D.L. C., N. M., T. B., "Modelling Crack Growth by Level Sets and the Extended Finite Element Method", *International Journal for Numerical Methods in Engineering*, vol. 51, p. 943-960, 2000.

[SUK 01] Sukumar N., Chopp D.-L., Moës N., Belytschko T., "Modeling holes and inclusions by level sets in the extended finite-element method", *Computer Methods in Applied Mechanics and Engineering*, vol. 190, p. 6183–6200, 2001.

[THE 98] Theocaris P.-S., Stravroulakis G., "Multilevel optimal design of composite strutures including materials with negative Poisson's ratio", *Structural Optimization*, vol. 15,, p. 8–15, 1998.

[TOS 06] Tosserams S., Etman L. F. P., Papalambros P.-Y., Rooda J.-F., "An augmented Lagrangian relaxation for analytical target cascading using the alternating direction method multipliers", *Structural Multidisciplinary Optimization*, vol. 31, p. 176–189, 2006.

[UME 05] Umesha P.-K., Venuraju M.-T., Hartmann D., Leimbach K.-R., "Optimal design of truss structures using parallel computing", *Structural and Multidisciplinary Optimization*, vol. 29, p. 285–297, 2005.

[VEN 03] Venter G., Sobieszczanski-Sobieski J., "Particle swarm optimization", *AIAA Journal*, vol. 41, num. 8, 2003.

[WAN 03] Wang M. Y., Wang X., Guo D., "A level-set method for structural topology optimization.", *Computer Methods in Applied Mechanics and Engineering*, vol. 192, p. 227–246, 2003.

[WIL 05] Wilke D. N., Analysis of the particle swarm optimization algorithm, Master's thesis, Department of Mechanical and Aeronautical Engineering, University of Pretoria, South Africa, 2005.